Sorteos para hacer justo un Mundial de fútbol

Pablo Laliena Bernués

Trabajo de fin del grado de Matemáticas

Universidad de Zaragoza

SUMMARY

The draw of the group phase for the 2014 World Cup of football, the most prestigious international tournament for national teams, was highly criticized for the injustices that it commited. These injustices were generated by the inefficient draw that the FIFA created in order to fulfill certain geographical constraints.

On the basis that a draw has to be at least fair and balanced (also equiprobable, but this was not the part in which the criticisms were centered) there were many countries that said they felt penalized. Because of this, the probabilist Julien Guyon wrote an article of huge international impact, *Rethinking the FIFA World Cup Final Draw* [3], where he put in doubt the goodness of that draw. At the same time he proposed other types of draws that would solve the flaws of the FIFA draw.
In this article Guyon proves that FIFA draw is not equiprobable (something essential at a mathematical level) but also agrees with countries such as Chile and USA who complained about having suffered negative treatment. These countries ended up in groups of a highly competitive level, at the same time that other countries such as France benefitted from the way the draw was carried out. To solve the problems of the FIFA draw Guyon proposed some possible alternative draws that will make the draw fairer and more balanced, in the sense that all the groups have a similar potential (competitive level). One of these draws is the same as the one that is done for the UEFA Champions league, but with a few modifications that make it equiprobable. The other one is a draw in which he orders the teams in 4 pots through a curve in the shape of an "S". After that he divides the principal draw in 2 independent draws and then applies a method analogous to the one used in the Champions draw.

The development of this work has been the following:
1) First of all I will start defining the concept of balancedness and fairness in a draw, and I will propose the search for the so called "Perfect" draw. For reasons that I will explain later, the potential of a group will be considered as the sum of the punctuations of the 3 strongest teams of each group. And a draw will be balanced if those competitive levels are similar. With this definition I will proceed to find a draw where the posibble solutions are all perfect in that sense, meaning that every group will have exactly the same potential. Furthermore, this draw will fulfill the geographical and seeding conditions imposed by FIFA.

To solve this problem I will pose a problem of binary linear programming whose solutions will be the "Perfect" draws. But the amount of variables and restrictions is too big and so the problem cannot be solved with the software I have. Another way to find all the solutions of that kind, is to make a program that will go through every possible way of ditributing the teams in 8 groups. But the amount of ways of ditributing the teams is so big that it makes this procedure non-viable in a reasonable amount of time. So what was finally done was to create an algorithm that only searches the solutions in certain combinations of teams in 4 pots (distributions). For this, starting on the basic distributions I will define the concept of non-valid change and a certain realation between the changes, in a way that the non-validity of a change will imply the non-validity of other changes (Definition 7 and Theorem 2,3). In this way I can eliminate the huge majority of ditributions, keeping only those which can generate the solutions of the problem. After finding those distributions I make some computer programs that give me the solutions in less than a minute. Once the solutions are found, that is to say, the way of

ditributing 32 teams in 8 groups fulfilling the FIFA's restrictions and making the sum of the punctuations of the 3 stronger teams of each group the same, I will propose 2 ways of making a draw that will decide which solution is the one that should be used in the group phase of the World Cup.

2) After creating the so called "Perfect" draw I see that it could have a little drawback: it is not really attractive for the public. For that reason I will pose another kind of draw that even if it loses the optimality of the "Perfect" draw it still gives us good results, at the same time that it is more visually attractive. This procedure will be heuristic in the sense that it will not be created as the solution of an optimization problem. I will simply look for a draw that gives us solutions as close as the optimal ones as possible through a heuristic procedure. For that I will analyze in detail the draw proposed by Guyon to try to find its weak points, that is to say, the reason that it still gives a relatively high variability (despite being lower than the FIFA by a huge margin) in the potential of the groups. Following this idea I will propose a second draw similar to the one made by Guyon (the second one commented) but which will give us better solutions. This draw will be equiprobable and will have an attractive implementation for the public. It will also fulfill FIFA's restrictions, and it will be called draw 323.

3) Once both draws have been created, I will proceed to compare them with Guyon's and FIFA's. For that I will simulate 10.000 draws of: Guyon, FIFA and 323. And I will take the population of the "Perfect" draw. After obtaining those samples I will compare them using dispersion measures: rank and variance. I will compare the results through a hypothesis test, numerical summaries and some graphs in which the difference between draws will be more clear.

4) After comparing the draws I will draw some conclusions based on the statistical results. Finally I will propose some possible expansions of this work.

Índice general

1. Introducción **7**

2. Búsqueda agrupaciones perfectas **13**
2.1. Función objetivo 13
2.2. Algoritmo de búsqueda de agrupaciones perfectas. 18

3. Creación de los sorteos. **23**
3.1. HEURÍSTICA DEL SORTEO 323 24

4. Análisis de resultados. **27**

5. Conclusiones. **33**
5.1. Ampliación 33

Bibliografía **35**

A. Programación dinámica **37**
A.1. Búsqueda de los posibles B4. 37

B. ANEXO II: Programas informáticos **41**
B.1. Sorteo FIFA 41
B.2. Sorteo de Guyon 49
B.3. Agrupaciones perfectas 54
B.4. Sorteo Perfecto 62
B.5. Sorteo 323 77

Capítulo 1

Introducción

El 6 de diciembre de 2013 se llevó a cabo el sorteo de la Copa del Mundo de fútbol de 2014, un sorteo que tuvo mucha repercusión a nivel internacional, no solo por su significado deportivo, sino por el aluvión de críticas que recibió. A raíz de esto, el probabilista Julien Guyon, escribió el artículo: Rethinking the FIFA World Cup final draw [3] que tuvo gran repercusión en los medios de comunicación (New York Times, El País,...). En él analizaba las deficiencias del sorteo de la FIFA [9] a la vez que presentaba otro tipo de sorteos que las solventaba. El objetivo de este trabajo será el de analizar dichos sorteos y crear otros sorteos que mejoren los propuestos en dicho artículo.
Para que un sorteo se consideré justo o bueno (en este caso el sortear el modo en el que 32 países son repartidos en 8 grupos) debe cumplir unos requisitos mínimos o principios básicos; estos son:
-Justicia: El sorteo no puede perjudicar ni beneficiar a ningún equipo, es decir, ningún equipo debería tener más posibilidades que otro de acabar en un grupo más duro o más flojo.
-Igualdad: El sorteo debería dar lugar a 8 grupos igualados, es decir, con similar potencial competitivo.
-Equiprobabilidad: Cualquiera de los posibles resultados a los que pueda dar lugar el sorteo han de tener la misma probabilidad. En términos matemáticos estos significa que la distribución de probabilidad sobre el conjunto de resultados es uniforme discreta.
Pasaré ahora a explicar el sorteo de la FIFA [9] y a comentar brevemente el porqué no cumple ninguno de los principios básicos antes nombrados (mas detalle al respecto se puede encontrar en el artículo de Guyon [3]). A la vez introduciré los rankings y el continente al que pertenecen, por comodidad se llamará continente a la zona de calificación.

BOMBO 1	BOMBO 2	BOMBO 3	BOMBO 4
Brasil(1,11,S)	Chile(12,12,S)	EEUU(13,13,N)	Holanda(9,8,E)
España(2,1,E)	Costa de Marfil(17,17,Af)	México(23,24,N)	Italia(10,9,E)
Alemania(3,2,E)	Ecuador(21,22,S)	Costa Rica(24,31,N)	Inglaterra(11,10,E)
Argentina(4,3,S)	Ghana(22,23,Af)	Honduras(27,34,N)	Portugal(14,14,E)
Colombia(5,4,S)	Algeria(25,32,Af)	Japón(28,44,As)	Grecia(15,15,E)
Bélgica(6,5,E)	Nigeria(26,33,Af)	Irán(29,49,As)	Bosnia Herze(16,16,E)
Uruguay(7,6,S)	Camerún(32,59,Af)	Corea Sur(30,56,As)	Croacia(18,18,E)
Suiza(8,7,E)		Australia(31,57,As)	Rusia(19,19,E)
			Francia(20,21,E)

Cuadro 1: Distribución de los bombos en el sorteo de la FIFA.

El paréntesis asociado a cada país significa los siguiente:
-La letra indica en que zona se clasificó cada país: E=Europa, S= América del Sur, N=América del Norte y Central, As=Asia, Af=África . Notar que ningún equipo de Oceania se clasificó, Australia competía con los países asiáticos.
-El primer número indica el rango que ocupa cada país dentro de los clasificados (según el ranking FIFA [10]), se considerará que el anfitrión, Brasil, tiene rango 1.
-El segundo número indica su posición en el ranking FIFA [10] en el momento del sorteo.

La FIFA además tiene 2 reglas (que no se modificarán) y que conviene tener presentes: La primera es que independientemente de la posición que ocupe en el ranking, el anfitrión de la Copa del Mundo será el cabeza de serie y los 8 primeros cabezas de serie estarán en diferentes grupos. La segunda es que al ser la Copa del Mundo uno de los pocos torneos intercontinentales, con el fin de beneficiar tales enfrentamientos se crean una serie de restricciones geográficas que impiden que en la fase de grupos los países de una región jueguen entre si. Por ello, no pueden coincidir en un mismo grupo los países sudamericanos, norte/centroamericanos, asiáticos y africanos. Y no puede haber más de dos europeos en el mismo grupo (el gran número de estos imposibilita poder exigir más).

Con el fin de proteger la restricción geográfica la FIFA crea los anteriores bombos, en el primero están los 8 cabezas de serie, en el segundo los africanos y sudamericanos restantes, en el tercero los norte/centroamericanos y los asiáticos y en el último los europeos restantes. Cada grupo obtendrá un equipo de cada bombo. Pasaré a explicar de forma esquemática como se realiza el sorteo:
Paso 1- Un equipo del bombo 4 es sorteado y pasa a formar parte del bombo 2.
Paso 2- Se sortean los equipos de bombo 1 y según van saliendo van entrando en el primer grupo vacío (Brasil va al grupo 1 ó A por ser el cabeza de serie).
Paso 3- Ayudándose de un bombo auxiliar se sortea al europeo del bombo 2 que irá a un grupo con un sudamericano.
Paso 4-Se sortea el bombo 2 y los equipos que van saliendo en dicho sorteo van a parar al primer grupo vacío (que solo tenga 1) admisible (respetando las restricciones geográficas y teniendo en cuenta el resto del sorteo del bombo).
Paso 5-Se sortea el bombo 3, los equipos que van saliendo van entrando al primer grupo vacío (que solo tenga 2). A continuación se hace lo mismo con el bombo 4

Pasaré ahora a poner el resultado de dicho sorteo:

Brasil	Camerún	México	Croacia	(74,112)-(42,53)
España	Chile	Australia	Holanda	(54,78)-(23,21)
Colombia	Costa de Marfil	Japón	Grecia	(65,80)-(37,36)
Uruguay	Italia	Costa Rica	Inglaterra	(52,56)-(28,25)
Suiza	Ecuador	Honduras	Francia	(76,84)-(49,50)
Argentina	Nigeria	Irán	Bosnia Herze.	(75,101)-(46,52)
Alemania	Ghana	EEUU	Portugal	(52,52)-(30,29)
Bélgica	Algeria	Corea del Sur	Rusia	(80,112)-(50,56)

Cuadro 2: El primer paréntesis indica: (Suma total primer número, Suma total segundo número). El segundo indica: (Suma 3 mejores primer número, suma 3 mejores segundo número).

Como se puede observar en el resultado del sorteo, no hay igualdad entre los grupos, las sumas deberían ser semejantes y vemos que hay diferencias de 10 e incluso de más de 20. Y además si uno se fija en el sorteo, se puede ver que los países; EEUU y Chile están predestinados a acabar en un grupo fuerte (cuanto mas pequeño es el número del ranking tanto más fuerte es el equipo asociado a este), puesto que además de ir con uno de los cabezas de serie irán con un europeo (situados todos en la parte media-alta de los ranking) y además no podrán ir ni con los asiáticos ni con los africanos respectivamente. Luego como era de esperar, ambos países acabaron en grupos duros. Además siguiendo esta misma linea de razonamiento se ve que tampoco es justo que se sortee el equipo que pasa del bombo 4 al 2, pues lo justo sería que el más débil fuera el que se cambiará, se está beneficiando a este equipo más débil al realizar dicho sorteo [2].
Luego el sorteo FIFA [6] ni es justo ni esta igualado, y además como prueba Guyon en su primera parte del artículo [3] tampoco es equiprobable. El problema de la equiprobabilidad está en la forma de sortear el segundo bombo, un problema que tendría fácil solución creando un segundo bombo auxiliar en el cuál estuvieran los europeos del primer bombo y cuyo objetivo sería que los sudamericanos fueran con los europeos, evitando así problemas con las restricciones geográficas y con la equiprobabilidad del sorteo.

Durante el resto del trabajo se usará el ranking del 1-32 (es decir el primer número asociado a cada país en el cuadro 1). La razón es la siguiente: El ranking de la FIFA [10] abarca desde el número 1 al número 59 y a la hora de intentar crear sorteos en los que los grupos tengan potencial semejante (este potencial vendrá marcado por el puesto en el ranking) será difícil trabajar con número tan dispares, por ello se usará el rango de los 32 clasificados, esto es: Brasil el 1 por ser el anfitrión y los demás en orden descendente según su posición en el ranking FIFA (España 2, Alemania 3,...). A la posición que un equipo ocupa en este nuevo ranking se le denominará peso y por tanto, a menor peso más fuerte es o más nivel competitivo tiene un equipo.

El sorteo final de Guyon, el mejor en términos de conseguir la igualdad y por tanto también el más justo es el siguiente :
Ordena a los equipos usando el rango de los 32 clasificados, en 4 bombos siguiendo una curva en forma de S. En el primero están los del 1-8, en el segundo del 16-9 en el tercero del 17-24 y en el cuarto del 32-25 (a partir de ahora se llamará a cada país por su posición en el ranking y no por su nombre). Y partiendo por la mitad estos bombos (el 32-25 implica que el bombo es 32-31-30...-25 y el 1-8 que es 1-2-3...-8) crea 2 sorteos independientes:

1(Brasil,S)	16E	17Af	32Af		5(Colombia,S)	12S	21S	28As
2(España,E)	15E	18E	31As		6(Bélgica,E)	11E	22Af	27N
3(Alemania,E)	14E	19E	30As		7(Uruguay,S)	10E	23N	26Af
4(Argentina,S)	13N	20E	29 As		8(Suiza,E)	9E	24N	25Af

Cuadro 3: La parte de la izquierda es un sorteo y la de la derecha otro.

Cada columna es un bombo y como antes cada grupo tendrá un equipo de cada bombo. La forma en que hace el sorteo es la siguiente:
Primero se le da nombre a los cabezas de serie (los 8 primeros) y se tiene en cuenta su continente, y a los demás equipos se les identifica con su continente (ignorando su posición en el ranking). La primera parte trata de buscar todas las posibles soluciones al sorteo que cumplan las restricciones

geográficas (no hay diferencia entre que el 15 o el 14 estén en un grupo, solo importa su continente) en cada sorteo. Una vez halladas todas las posibles se procede a sortear una de ellas en cada sorteo (cada solución ha de tener la misma probabilidad de salir, como ya se explicará) y tras saber que solución geográfica regirá cada sorteo se proceden a sortear que número es asignado a cada letra (estando el número relacionado con la letra y obviamente formando parte del bombo correspondiente). En el caso de que no hubiera ninguna solución geográfica se intercambiarían 2 equipos de cada bombo 4. Hay varías maneras de hacer esto, en el momento que en cada sorteo hubiera al menos una solución geográfica se usaría dicho sorteo. Si esto no funciona se haría el mismo método con el bombo 3, 2 y 1. Y en el caso de que esto tampoco funcionara se procedería hacer ese mismo tipo de cambios pero en varios bombos a la vez (para más detalle ver [3]).

El hecho de que cada solución geográfica tenga la misma probabilidad de salir dentro de cada sorteo, se debe a que todas tienen el mismo número de soluciones, es decir, que si en cada solución geográfica buscaremos todas las formas de repartir los equipos se vería que todas tienen el mismo número de formas. Luego el sorteo será equiprobable (demostración completa en el anexo de Guyon). Pondremos ahora un ejemplo de este sorteo:

Brasil, 1S	E, 16	E,19	31,As		Colombia,5S	E,11	N,23	Af,26
España, 2E	N, 13	E,18	32,Af		Bélgica, 6E	E,9	S,21	Af,25
Alemania, 3E	E,15	Af,17	29,As		Uruguay,7S	E,10	N,24	As,28
Argentina,4S	E,14	E,20	30,As		Suiza,8E	S,12	Af,22	N,27

Primero se sortearía la solución geográfica (las letras E, N, Af...) y después se sortearía a los equipos que pueden ocupar ese sitio (por ejemplo, Brasil va con E,16 pero ha sido el 16 como podría haber sido el 15 o el 14, la segunda parte del sorteo ha dictaminado que fuera el 16).
Vemos como la suma de los tres más fuertes de cada grupo y de los cuatro ahora es más pareja (solo se mirará esto usando el ranking 1-32):
Peso de los 3 más fuertes: 36-33-35-38-39-36-41-42
Peso de los 4: 67-65-64-68-65-61-69-69
En este ejemplo concreto, se ve que los resultados son muy buenos, en la suma de los 3 más fuertes el rango es de 9 y en la suma de 4 es de 8. Mucho mejor que los grupos que dio el sorteo de la FIFA. Pero, ¿será esto una coincidencia, o realmente existe esta atronadora diferencia en los sorteos? Más adelante se hará un estudio estadístico de todos los sorteos, donde se comprobará o no si en realidad el sorteo de la FIFA es tan desigual y el sorteo de Guyon es tan igualado.

Aunque se ha hablado de igualdad de grupos y se ha comentado, no se ha terminado de especificar su definición. Como ya se ha dicho Guyon trabaja con 2 posibles formas de entenderla, centrándose más en una, estas son: que la suma de los tres equipos menos pesados de cada grupo sea lo más pareja posible y que la suma de los cuatro equipos de cada grupo sea lo más pareja posible. Por la forma en que ha enfocado su artículo a nivel de resultados y por el tipo de sorteos que crea, se ve que él se centra más en la segunda forma de entenderla.

En los sorteos que crearé y que intentarán mejorar (desde un punto de vista estadístico) el sorteo de Guyon y de la FIFA, a diferencia de Guyon me centrare más en la primera forma de entender la igualdad: Conseguir que la suma de los tres equipos menos pesados (o más fuertes) de cada grupo sea

lo más pareja posible. Es decir, valoraré más el hecho de que en los grupos que salgan en un sorteo la suma de los 3 equipos más fuertes de cada grupo sea lo más pareja posible a que lo sea la de 4, aunque en la medida de lo posible trataré que la suma de 4 no quede desnivelada, pero se pondrá siempre por delante a la suma de 3. El porqué de mi elección se basa en lo que se entiende por "Grupo de la muerte" .
Cuando en fútbol se habla de "Grupo de la muerte" se habla de un grupo en el que uno de los que era más o menos favorito (entre los 14 mejores) va a ser eliminado. Esto ocurre en el caso del mundial cuando hay 3 equipos fuertes en un mismo grupo, puesto que al clasificarse únicamente los 2 primeros uno quedará eliminado, este grupo se denomina así independientemente de la fuerza del cuarto equipo. Luego siguiendo este razonamiento mi objetivo será erradicar la existencia de los "Grupos de la muerte" en la medida de lo posible (también el contario, es decir el que haya 3 equipos muy débiles en un grupo y que por ende uno se valla a clasificar, aunque normalmente si se da uno se dará el otro), intentando para ello minimizar en lo máximo posible la diferencia de la suma de los pesos de los 3 primeros.

Por lo tanto como lo que se quiere es intentar que la suma de los tres primeros sea lo más pareja posible, mí primera idea será conseguir la igualdad perfecta, es decir un sorteo en el que cada posible solución sean grupos en los que la suma de los 3 primeros sea exactamente la misma en cada grupo , usaré el ranking 1-32 de la manera ya descrita. Este sorteo será denominado como: Sorteo "Perfecto" y cada una de sus soluciones será denominada: agrupación perfecta. Procedemos ahora a intentar buscar un sorteo de estas características y que además sea equiprobable, cumpla las restricciones geográficas y haga que los 8 primeros cabezas de serie vayan a distintos grupos y el anfitrión (Brasil) sea el cabeza de serie número 1.

Para crear dicho sorteo primero necesitaré encontrar dichas soluciones. Este problema está relacionado con el problema de partición de conjuntos. En ese problema, se dispone de un conjunto (o multiconjunto) de enteros S y se desea separarlos en grupos en los que la suma sea la misma. Véase, por ejemplo, [6] y [7]. Lo que diferencia nuestro problema con el de partición de conjuntos es que en el nuestro ya hay fijado un elemento en cada grupo (cabezas de serie) y que además, no se quiere que la suma de todos los elementos sea igual sino la suma de los 3 elementos con menos valor. Por tanto daré una solución original a nuestro problema.

Una vez creado el sorteo "Perfecto" plantearé otro modelo de sorteo (323) que aunque no dará agrupaciones perfectas seguirá mejorando el propuesto por Guyon. La razón de ser de este nuevo sorteo es que el sorteo Perfecto será poco vistoso a la hora de llevarlo a la práctica, mientras que el de tipo 323 tendrá un formato similar al de la FIFA que se difunde por televisión a todo el mundo.

A continuación procederé a encontrar las agrupaciones perfectas con las que crearé el denominado sorteo "Perfecto".

Capítulo 2

Búsqueda agrupaciones perfectas

2.1. Función objetivo

Ya se ha expuesto cómo se quiere que sea nuestro denominado sorteo perfecto, pero para poder realizarlo debo encontrar todas las agrupaciones perfectas. Como lo que en realidad se quiere es minimizar el rango de la suma de los tres primeros, en lo primero en que se piensa es en programación binaria lineal. Procederé por tanto a atacar este problema de esta manera.

Notar que se pueden fijar a los 8 primeros cabezas de serie cada uno en un grupo (debido a la restricción FIFA), el cabeza de serie 1 irá al grupo 1, el cabeza de serie 2 irá al grupo 2, etc. , pero no así con ningún otro equipo. Pasaré por tanto a construir una función objetivo que minimice el rango de la suma de 3 equipos, es decir, minimizar la diferencia de peso existente entre el grupo más pesado y el menos pesado. Iré declarando las variables y poniendo las restricciones conforme se vayan necesitando. De ahora en adelante $i = 1,...,24$ indicará el equipo, $j = 1,...,8$ indicará el grupo. Por comodidad usaré la siguiente notación: $C_i = i + 8 \quad (i = 1,...,24)$ Peso del equipo i.

VARIABLE:

$$X_{ij} = \begin{cases} 1 & \text{si el equipo i está en el grupo j} \\ 0 & \text{si el equipo i no está en el grupo j} \end{cases}$$

$(i = 1,...,24)x(j = 1,...,8)$

RESTRICCIONES:

$$\begin{cases} \sum_{i=1}^{24} X_{ij} = 3 \quad (j = 1,...,8) \text{ A cada grupo entran 3 equipos más} \\ \sum_{j=1}^{8} X_{ij} = 1 \quad (i = 1,...,24) \text{ Cada equipo va a un solo grupo} \end{cases}$$

-Europa (Máximo 2 por grupo):

$X_{1j} + X_{2j} + X_{3j} + X_{6j} + X_{7j} + X_{8j} + X_{10j} + X_{11j} + X_{12j} \leq 2 \quad (j = 1,4,5,7)$

$X_{1j} + X_{2j} + X_{3j} + X_{6j} + X_{7j} + X_{8j} + X_{10j} + X_{11j} + X_{12j} \leq 1 \quad (j = 2,3,6,8)$

-Sudamérica (Máximo 1 por grupo):

$X_{4j} + X_{13j} \leq 1 \quad (j = 2,3,6,8)$

$X_{4j} + X_{13j} \leq 0 \quad (j = 1,4,5,7)$

-África, Norteamérica y Asia (Máximo 1 por grupo):

$X_{9j} + X_{14j} + X_{17j} + X_{18j} + X_{24j} \leq 1 \quad (j = 1,...,8)$

$X_{5j} + X_{15j} + X_{16j} + X_{19j} \leq 1 \quad (j = 1,...,8)$

$X_{20j} + X_{21j} + X_{22j} + X_{23j} \leq 1 \quad (j = 1, ..., 8)$

VARIABLE:

$$Y_{ij} = \begin{cases} 1 & \text{si el equipo i está en el grupo j y es el equipo más pesado} \\ 0 & \text{si el equipo i no está en el grupo j o está y no es el equipo más pesado} \end{cases}$$

$(i = 1, .., 24) x (j = 1, ..., 8)$

RESTRICCIONES:

$X_{ij} - \sum_{k=i+1}^{24} X_{kj} \leq Y_{ij} \quad (j = 1, .., 8) x (i = 1, .., 24)$

$\sum_{i=1}^{24} Y_{ij} = 1 \quad (j = 1, .., 8)$ Solo hay un equipo más pesado en cada grupo.

VARIABLE:

$$Z_j = \begin{cases} 1 & \text{si el grupo j es el más pesado (solo cuentan los 3 equipos menos pesados)} \\ 0 & \text{si el grupo j no es el más pesado} \end{cases}$$

$$W_j = \begin{cases} 1 & \text{si el grupo j es el menos pesado (solo cuentan los 3 equipos menos pesados)} \\ 0 & \text{si el grupo j no es el menos pesado} \end{cases}$$

$j = 1, ..., 8$

RESTRICCIONES:

$\left(h + \sum_{i=1}^{24} C_i (X_{ih} - Y_{ih}) \right) - \left(j + \sum_{i=1}^{24} C_i (X_{ij} - Y_{ij}) \right) \geq (Z_h - 1) 200$

$(h = 1, ..., 8) x (j = 1, ..., 8 \setminus h)$

$\sum_{j=1}^{8} Z_j = 1$ (solo un grupo puede ser el más pesado)

$\left(h + \sum_{i=1}^{24} C_i (X_{ih} - Y_{ih}) \right) - \left(j + \sum_{i=1}^{24} C_i (X_{ij} - Y_{ij}) \right) \leq (1 - W_h) 200$

$(h = 1, ..., 8) x (j = 1, ..., 8 \setminus h)$

$\sum_{j=1}^{8} W_j = 1$ (Solo un grupo puede ser el menos pesado).

VARIABLE:

$$M_i = \begin{cases} 1 & \text{si el equipo i está en el grupo más pesado y él no es el equipo más pesado} \\ 0 & \text{resto} \end{cases}$$

$$N_i = \begin{cases} 1 & \text{si el equipo i está en el grupo menos pesado y él no es el equipo más pesado} \\ 0 & \text{resto} \end{cases}$$

$i = 1, ..., 8$

RESTRICCIONES:

$X_{ij} + Z_j - Y_{ij} \leq M_i + 1 \quad (j = 1, ..., 8) x (i = 1, ..., 24)$

$\sum_{i=1}^{24} M_i = 2$ (2 equipos cumplirán esto, ya que el cabeza de serie es el menos pesado de cada grupo).

$X_{ij} + W_j - Y_{ij} \leq N_i + 1 \quad (j = 1,...,8)x(i = 1,...,24)$
$\sum_{i=1}^{24} N_i = 2$ (2 equipos cumplirán esto, ya que el cabeza de serie es el menos pesado de cada grupo).

Y la FUNCIÓN OJETIVO a minimizar quedaría de la siguiente manera:

$$Min\left(\sum_{i=1}^{24} C_i(M_i - N_i) + \sum_{j=1}^{8} j(Z_j - W_j)\right)$$

Este sería el planteamiento inicial del problema de programación entera, pero hay un problema, hay un número enorme de variables y restricciones:
Total variables: $192 + 192 + 8 + 8 + 24 + 24 = 448$
Total restricciones: $8 + 24 + 8 + 8 + 8 + 8 + 8 + 192 + 8 + 56 + 1 + 56 + 1 + 192 + 1 + 192 + 1 = 772$
Estos números se podrían reducir un poco teniendo en cuenta que hay valores que se pueden deducir matemáticamente (como que $Y_{1j} = 0 = Y_{2j} \quad (j = 1,...,8)$) y se podrían ahorrar cerca de 50 variables e incluso más, pero aun con todo este número seguiría siendo demasiado grande. Se debe simplificar este problema de alguna manera.
Usaré para ello un algoritmo aplicable a problemas de programación lineal binaria; el algoritmo de Balas [8]. Para aplicar este algoritmo son necesarias 2 cosas: Que la función objetivo sea lineal y que los coeficientes de la función objetivo sean positivos, este último punto no es un problema pues haciendo unos cambios de variable se puede conseguir que todos los coeficientes de la función objetivo sean positivos. El algoritmo consiste en lo siguiente (se explicará su aplicación a nuestro caso):

Paso 1: Tras conseguir que la función objetivo este formada únicamente por coeficientes positivos, estos se reordenan dentro de la misma función de menor coeficiente a mayor, es decir los que tengan coeficiente 0 (también hay que tenerlos en cuenta) van al principio, y aquellos coeficientes que sean más altos irán al final. Y se toma como cota inicial $= \infty$ (si se tiene una cota mejor se pude y debe poner, agilizará el algoritmo).
Paso 2: Cómo asignar valores a las variables:
Suponer que se tiene que asignar j 1s entre las variables $(X_1,...,X_n)$. El primer paso sería:
$X_1 = ... = X_j = 1$ el siguiente paso sería: $X_1 = ... = X_{j-1} = 1, X_j = 0, X_{j+1} = 1$ y así hasta llegar a la variable con mayor coeficiente. El siguiente paso llegado ese punto sería: $X_1 = ... = X_{j-2} = 1, X_{j-1} = 0, X_j = X_{j+1} = 1$ llamaré a este paso reiniciar las variables. La siguiente vez que se reinicie será: $X_1 = ... = X_{j-2} = 1, X_{j-1} = X_j = 0, X_{j+1} = X_{j+2} = 1$ y el 1 que se movería sería el de X_{j+2}. Y cuando haciendo esto se llegara al final, la siguiente vez que se reiniciara sería: $X_1 = ... = X_{j-3} = 1, X_{j-2} = 0, X_{j-1} = X_j = X_{j+1} = 1$. Ver [8].
Paso 3: Ramificación: Tras dar valores a las variables se calcula el valor de la función objetivo (independientemente de que cumplan las restricciones o no) si este valor es menor que la cota inicial se comprueba si cumple las restricciones y en caso positivo dicho valor será la nueva cota. Y además se reiniciarán las variables puesto que al hacer el siguiente paso de dar valores, el valor de la función objetivo será mayor o igual, luego no mejorará nada. A su vez si el valor de la función objetivo es mayor o igual que la cota se reiniciarán las variables puesto que al hacer el siguiente paso el valor de la función objetivo seguirá siendo mayor o igual que el valor anterior y por tanto también del valor de la cota.
Paso 4: Seguir dando valores a las variables hasta que:

a) Se haya llegado al final.
b) Aunque reinicies las variables el valor de la función objetivo es mayor que la cota, lo que implicará que no se va a mejorar el valor de la función objetivo.
Notar que este algoritmo da una única solución, si se quisieran encontrar todas habría dos formas:
a)Rechazar casos únicamente cuando se este seguro que el valor de la función objetivo será mayor estricto que la cota mínima.
b)Conforme se van obteniendo soluciones se van añadiendo restricciones que impidan obtener esa solución y se aplica el algoritmo al nuevo problema.
Notar que una de las ventajas del algoritmo de Balas es que en ningún momento exige la linealidad de las restricciones, las usa como comprobaciones de que los valores de las variables son correctos.

Readaptaré ahora las restricciones al hecho de que no hace falta que sean lineales y si el número de variables pasa a ser un número razonable con el que trabajar se intentará aplicar el algoritmo. El subíndice i, j sigue indicando lo mismo, al igual que el significado y valor de C_i.
Cómo las variables $X_{ij}, Z_j, W_j, M_i, N_i$ tienen el mismo significado, solo pondré sus restricciones asociadas .

FUNCIÓN OBJETIVO:

$$Min\left(\sum_{i=1}^{24} C_i(M_i - N_i) + \sum_{j=1}^{8} j(Z_j - W_j)\right)$$

RESTRICCIONES:

$$\begin{cases} \sum_{i=1}^{24} X_{ij} = 3 \quad (j = 1,...,8) \text{ A cada grupo entran 3 equipos más.} \\ \sum_{j=1}^{8} X_{ij} = 1 \quad (i = 1,...,24) \text{ Cada equipo va a un solo grupo.} \end{cases}$$

-Europa (Máximo 2 por grupo):

$$X_{1j} + X_{2j} + X_{3j} + X_{6j} + X_{7j} + X_{8j} + X_{10j} + X_{11j} + X_{12j} \leq 2 \quad (j = 1,4,5,7)$$
$$X_{1j} + X_{2j} + X_{3j} + X_{6j} + X_{7j} + X_{8j} + X_{10j} + X_{11j} + X_{12j} \leq 1 \quad (j = 2,3,6,8)$$

-Sudamérica (Máximo 1 por grupo):

$$X_{4j} + X_{13j} \leq 1 \quad (j = 2,3,6,8)$$
$$X_{4j} + X_{13j} \leq 0 \quad (j = 1,4,5,7)$$

-África, Norteamérica y Asia (Máximo 1 por grupo):

$$X_{9j} + X_{14j} + X_{17j} + X_{18j} + X_{24j} \leq 1 \quad (j = 1,...,8)$$
$$X_{5j} + X_{15j} + X_{16j} + X_{19j} \leq 1 \quad (j = 1,...,8)$$
$$X_{20j} + X_{21j} + X_{22j} + X_{23j} \leq 1 \quad (j = 1,...,8)$$

VARIABLE:

$$Y_i = \begin{cases} 1 & \text{el equipo i es el más pesado de su grupo} \\ 0 & \text{si el equipo i no es el más pesado de su grupo} \end{cases}$$
$(i = 1,...,24)$

RESTRICCIONES:

$$X_{ij} - \sum_{k=i+1}^{24} X_{kj} \leq Y_i \quad (j = 1,..,8)x(i = 1,..,24)$$

$\sum_{i=1}^{24} Y_i = 8$ (Hay 8 equipos más pesados, uno por cada grupo).

$$\left(h+\sum_{i=1}^{24} C_i X_{ih}(1-Y_i)\right)-\left(j+\sum_{i=1}^{24} C_i X_{ij}(1-Y_i)\right)\geq (Z_h-1)200$$
$(h=1,...,8)x(j=1,...,8\setminus h)$

$\sum_{j=1}^{8} Z_j = 1$ (Solo un grupo puede ser el más pesado).

$$\left(h+\sum_{i=1}^{24} C_i X_{ih}(1-Y_i)\right)-\left(j+\sum_{i=1}^{24} C_i X_{ij}(1-Y_i)\right)\leq (1-W_h)200$$
$(h=1,...,8)x(j=1,...,8\setminus h)$

$\sum_{j=1}^{8} W_j = 1$ (Solo un grupo puede ser el menos pesado).

$\sum_{j=1}^{8} X_{ij} Z_j (1-Y_i) \leq M_i \quad (i=1,...,24)$

$\sum_{i=1}^{24} M_i = 2$ (2 equipos cumplirán esto, ya que el cabeza de serie es el menos pesado de cada grupo).

$\sum_{j=1}^{8} X_{ij} W_j (1-Y_i) \leq N_i \quad (j=1,...,8)x(i=1,...,24)$

$\sum_{i=1}^{24} N_i = 2$ (2 equipos cumplirán esto, ya que el cabeza de serie es el menos pesado de cada grupo).

Total variables: $192+24+8+8+24+24=280$

Total restricciones: $8+24+40+192+1+57+57+25+25=429$

A pesar de haber reducido muy notablemente el número de variables y el número de restricciones (166 y 333 reducidas respectivamente) no se va a poder resolver el problema por este método, debido a que el número de variables es demasiado grande. Se podría pensar en usar las igualdades que se tienen en las restricciones para quitar variables, pero eso acabaría perjudicando, pues se dejaría de saber cuantos 1s se pueden repartir.

Voy a ver que ocurriría: Cómo se tiene que: $\sum_{j=1}^{8} X_{ij} = 1$, se puede hacer $X_{i1} = 1 - \sum_{j=2}^{8} X_{ij}$. Pero asi como con la primera forma se tenían 8 variables (por cada i) y se sabía que una de ellas era 1, al sustituir como en la segunda parte no se puede asegurar que el 1 este en $X_{2j}...X_{8j}$ y por tanto esto complicaría el aplicar el algoritmo ya que habría que aplicarlo varías veces según el número de 1s a repartir (de menor cantidad a mayor).

La razón de tener demasiadas variables se ve claro cuando se calcula una aproximación de las formas posibles que tenemos de dar valores 1 a las variables. Aún sabiendo que hay que dar a 38 de ellas el valor 1; el número total de formas de dar valor 1 a las variables es de:
$\binom{280}{38} = \frac{280!}{38!242!} \simeq 1{,}351 \cdot 10^{47}$. Y con este número de posibles soluciones no se puede trabajar, pues aun en el hipotético caso de que el algoritmo de Balas fuera un algoritmo que funciona de una manera extraordinaria de tal forma que la ramificación consiguiera que solo se tuviera que comprobar $\frac{1}{1{,}000{,}000{,}000}$ de todas las posibles soluciones, se tendría que seguir pasando por unas 10^{38} soluciones un número tremendamente grande. Luego va a ser imposible resolver el problema de esta manera. Se tendrá que buscar una forma alternativa de resolverlo.

2.2. Algoritmo de búsqueda de agrupaciones perfectas.

Tras la imposibilidad de encontrar las agrupaciones perfectas usando programación binaria lineal, pasamos a enfocar el problema de otra manera. Recordaré cuál era este:
El problema que se tiene que solucionar es el siguiente: Dada una lista de 32 equipos a los que asociaré un peso del 1-32 respectivamente según su ordenación en un ranking, en este caso el ranking FIFA, el objetivo es repartir estos 32 equipos en 8 grupos de 4 equipos cada uno. Además se exige que los 3 equipos con menos peso de cada grupo sumen lo mismo, esto no se puede conseguir si se tomaran los equipos del 1-24 ya que la media sería 37.5; se exigirá por tanto que la suma sea 38. Además se debe tener en cuenta la restricción impuesta por la FIFA de que los equipos con peso del 1-8 no pueden ir en el mismo grupo. Estos equipos son los llamados cabezas de serie.
El objetivo será hallar un algoritmo de resolución del problema que asegure que se encontrarán todas las soluciones de dicho tipo y que sea lo más simple posible. De ese modo creando los programas informáticos necesarios se hallarán todas las soluciones en un tiempo razonable. A la hora de crear este algoritmo obviaré las restricciones geográficas, pero no a la hora de aplicarlo a un caso concreto. Para ello empezaré dando las siguientes definiciones:

DEFINICIÓN 1: Se llamará **agrupación** a cualquier 8-tupla de elementos de $\{1,...,32\}^4$; $((i_1,j_1,k_1,l_1),...,(i_8,j_8,k_8,l_8))$ tal que $i_1,...,l_8$ son diferentes entre sí, y: $i_m < j_m < k_m < l_m$ para todo $m = 1,...,8$ y $i_m < i_{m1}$ si $m < m1$.

NOTA 1: Ver que de esta forma se evita repetir resultados, puesto que el grupo desordenado 2-22-17-28 es el mismo que la agrupación 2-17-22-28. A partir de ahora los grupos desordenados dejarán de tenerse en consideración.

DEFINICIÓN 2: Se llamará **solución** a toda aquella agrupación que cumpla : $i_m + j_m + k_m = 38$, con $m = 1,...,8$.

Como los cabezas de serie no pueden coincidir en un grupo, fijaré a cada uno en un grupo quedando así por repartir 24 equipos. La primera idea intuitiva sería crear un programa informático que creara todas las posibles agrupaciones y devolviera las que fueran solución. Esta opción es inviable puesto que el número total de agrupaciones que tendría que comprobar serían: $\frac{24!}{(3!)^8} \simeq 3,6 \cdot 10^{17}$ (el divisor se debe a que no todas son agrupaciones). Sabiendo que un ordenador normal hace del rango del 10^9 operaciones por segundo, dicho programa tardaría unos 4.000 días, algo inviable. Habrá que enfocar el problema de otra manera.

Como se quiere repartir 32 equipos en 8 grupos tiene sentido pensar en crear 4 bombos de 8 equipos cada uno, de forma que cada grupo tendrá un equipo de cada bombo.

DEFINICIÓN 3: Se llamará **distribución** a una 4-tupla (B1,B2,B3,B4) tal que cada B_i estará formado por 8 números del 1-32 distintos y además $BI \cap BJ = \emptyset$ si $I \neq J$ con $I,J = 1,2,3,4$.

DEFINICIÓN 4: Se llamarán **casos** generados por una distribución (B1,B2,B3,B4) a las agrupaciones $((i_1,j_i,k_1,l_1),...,(i_8,j_8,k_8,l_8))$ tal que $i_m \in B1$, $j_m \in B2$, $k_m \in B3$, $l_m \in B4$ con $m = 1,...,8$.

NOTA 2: Cada agrupación viene generada por una sola distribución. En efecto, dada una agrupación: $((i_1,j_1,k_1,l_1),...,(i_8,j_8,k_8,l_8)))$, ésta será un caso generado por la distribución: $B1 = \{i_1,...,i_8\}$, $B2 = \{j_1,...,j_8\}$,$B3 = \{k_1,...,k_8\}$,$B4 = \{l_1,...,l_8\}$. Y por tanto cada solución del problema tam-

bién vendrá generada por una sola distribución. Por la restricción de la FIFA sobre los cabezas de serie:$B1 = \{1,...,8\}$, ya que de otra forma no se cumplirá dicha restricción. Es decir: $i_m = m$ con $m = 1,..,8$.

Con el fin de simplificar el problema procederé ahora a identificar todos los posibles B4 que creen una distribución que pueda generar soluciones. Por la definición de soluciones y el hecho de que cada una de estas es un caso generado por una cierta distribución, la suma del peso de los equipos de B1, B2, B3 ha de ser: $38 \times 4 = 304$. Es decir la suma de los pesos de los equipos de B4 ha de ser: $528 - 304 = 224$, el total menos la suma de los pesos del resto de bombos.

LEMA 1. *Los B4 cuya suma de los pesos de sus equipos es 224 y que por tanto forma una distribución conjunto a B1,B2,B3 que puede generar alguna solución son los siguientes:*

24	23	22	21	23
25	25	25	26	24
26	26	27	27	27
27	28	28	28	28
29	29	29	29	29
30	30	30	30	30
31	31	31	31	31
32	32	32	32	32
B4A	**B4B**	**B4C**	**B4D**	**B4E**

Cuadro 2.1: Únicos B4 que pueden formar parte de una distribución que genere soluciones.

Demostración: Se hará en el anexo usando programación dinámica.

De la forma en que se ha ido procediendo es claro que cualquier distribución en la cuál B4 no sea uno de los anteriores no generará ninguna solución.
Tras haber hallado todos los B4 posibles y saber quién es B1, el siguiente paso lógico es intentar hallar todos los B2-B3 que conjunto a B4 y B1 me den todas las distribuciones que puedan generar alguna solución del problema. Por el momento lo que se tiene es que:
$B1 = \{1,2,3,4,5,6,7,8\}$
$B4 = a$ con $a \in \{B4A, B4B, B4C, B4D, B4E\}$
$B2, B3 \subset \{9,10,...,32\} \setminus B4$ tal que $\#B2 = \#B3$ y $B2 \cap B3 = \emptyset$
A partir de ahora cuando se hable de distribución esta seguirá la estructura que se acaba de mencionar.
A raíz de las definiciones de solución y de casos generados por una distribución se crea la siguiente definición:

DEFINICIÓN 5: Diremos que una distribución (B1,B2,B3,B4) es **no válida** si cumple alguna de las siguientes condiciones:
i) $8 + max\{j \in B2 \text{ t.q } j < min\{k \in B3\}\} + min\{k \in B3\} < 38$
ii) $1 + max\{j \in B2\} + min\{k \in B3 \text{ t.q } k > max\{j \in B2\}\} > 38$
iii) $\{j \in B2 \text{ t.q } j < min\{k \in B3\}\} = \emptyset$.

iv) $\{k \in B3 \text{ t.q } k > max\{j \in B2\}\} = \emptyset$.
Por convenio el máximo y mínimo del conjunto vacio se tomará igual a 0.

Teorema 1. *En una distribución* ***no válida*** *el conjunto de soluciones generadas por dicha distribución es el vacio.*

Demostración: El hecho de que una distribución cumpla alguna de las anteriores condiciones implica que:

En el caso de que i) sea cierto, esto implica que ninguno de los casos generados por dicha distribución será solución. En efecto, de todas las agrupaciones generadas, el grupo con más peso en el que puede estar $min\{k \in B3\}$ es: $8 + max\{j \in B2 \text{ t.q } j < min\{k \in B3\}\} + min\{k \in B3\}$ y si el peso de este grupo es menor estricto que 38, cualquier grupo en el que este $min\{k \in B3\}$ también tendrá peso menor estricto que 38 y por tanto dicha distribución no podrá generar soluciones. De manera análoga se sigue el caso de que ii) sea cierto, solo que en este caso lo que implica la condición es que dentro de todas las agrupaciones generadas, el grupo con menos peso en el que puede estar $max\{j \in B2\}$ tiene peso mayor estricto que 38. Luego por argumentación análoga a la anterior dicha distribución no podrá dar lugar a ninguna solución.

En el caso de que iii) sea cierto, esto implica que el conjunto de casos generados por dicha distribución es el vacio. Esto se debe a que el mínimo número de B3 es menor que todos los que hay en B2 y por ende será imposible que se genere ninguna agrupación, ya que ocurrirá que: $j_m > k$ con $k = min\{k \in B3\}$ para cualquier $j_m \in B2$. En el caso de que iv) sea cierto se sigue de manera análoga, solo que en este caso el máximo de B2 será mayor que todos los elementos de B3 y por tanto será imposible que se genere ninguna agrupación.
Falta por tanto hallar todos los B2-B3 que unidos a B1 y B4 con la estructura antes mencionada darán todas las posibles distribuciones que puedan generar soluciones. Todos estos B2-B3 se podrán generar haciendo cambios sobre cada distribución básica.

DEFINICIÓN 6: Una distribución se denominará básica si: $j < k \; \forall j \in B2, \; k \in B3$.

NOTA 3: Fijados los B1 y B4 solo hay una distribución básica. Ejemplo:

1	2	3	4	5	6	7	8	**B1**
9	10	11	12	13	14	15	16	**B2**
17	18	19	20	21	22	23	28	**B3**
24	25	26	27	29	30	31	32	**B4**

Cuadro 2.2: Esta sería la distribución básica si $B4 = B4A$. Recordar que siempre se toma $B1 = \{1,...,8\}$.

DEFINICIÓN 7: Dada una distribución (B1,B2,B3,B4) se llamará cambio de k equipos a $((i_1, j_1),...,(i_k, j_k))$con $i_m \in B2$ y $j_m \in B3$ con $m = 1,...,k$ y además $i_1 \neq ... \neq i_k \neq j_1 \neq ... \neq j_k$. Y se llamará distribución resultante del cambio $((i_1, j_1),...,(i_j, j_k))$ sobre (B1,B2,B3,B4) a
$(B1^{'}, B2^{'}, B3^{'}, B4^{'})$ con: $B1^{'} = B1$, $B4^{'} = B4$, $B2^{'} = B2 \cup \{j_1,...,j_k\} \setminus \{i_1,...,i_k\}$ y
$B3^{'} = B3 \cup \{i_1,...,i_k\} \setminus \{j_1,...,j_k\}$.
Se dice que un **cambio** sobre cierta distribución es **no válido** si la distribución resultante es no válida.

NOTA 4: Que los i_k sean diferentes a los j_k significa que los k cambios se hacen a la vez. Como

mucho se podrán hacer 8 cambios. Ver que un cambio de k equipos se puede ver como k cambios 1 hechos sobre las distribuciones que van resultando.

Un ejemplo de cambio de 1 equipo sería el cambio (16,17) sobre la distribución básica de B4A :

9	10	11	12	13	14	15	16	**B2**
17	18	19	20	21	22	23	28	**B3**
9	10	11	12	13	14	15	**17**	$B2^{'}$
16	18	19	20	21	22	23	28	$B3^{'}$

Cuadro 2.3: Este sería el B2-B3 inicial y el $B2^{'} - B3^{'}$ resultantes, B1 y B4 no los pongo porque no varían y ya se han descrito anteriormente.

DEFINICIÓN 8: Dados dos cambios de 1 equipo (i,j), (i1,j1) hechos sobre la misma distribución. Se dirá que el cambio (i1,j1) es más fuerte que el cambio (i,j) si se cumplen las siguientes condiciones:
i)$i = i1$ ó $j = j1$
ii)$j1 - i1 > j - i$

LEMA 2. *Dado el cambio de 1 equipo (i,j) sobre una distribución básica se tiene que la distribución resultante (B1,B2,B3,B4) verifica :* $max\{k \in B2\} = j$ *y* $min\{k \in B3\} = i$

Demostración: Evidente por definición de distribución básica.

Teorema 2. *Si la distribución resultante del cambio 1* (i, j) *sobre una distribución básica es no válido, la distribución resultante de cualquier cambio más fuerte que el cambio* (i, j) *sobre la misma distribución será también no válida.*

Demostración: Sea (i1,j1) un cambio más fuerte que (i,j). Y sea $B2^{'}, B3^{'}$ la distribución resultante del cambio (i1,j1)(B1 y B4 permanecen invariantes) y B2,B3 la del cambio (i,j). Por la definición 5, si el cambio (i,j) es no válido cumplirá alguna de las siguientes condiciones:

i) Si el cambio (i,j) es no válido por cumplir i) esto significa que (Lema 2):
$8 + max\{k \in B2 \text{ t.q } k < i\} + i < 38$. Si el cambio más fuerte es el cambio (i,j1) con j1>j, dicha desigualdad seguirá siendo cierta con $B2^{'}$ puesto que $max\{k \in B2^{'} \text{ t.q } k < i\} = max\{k \in B2 \text{ t.q } k < i\}$ ya que j,j1 > i. Si el cambio más fuerte es (i1,j) con i1<i:
$8 + max\{k \in B2^{'} \text{ t.q } k < i1\} + i1 < 8 + max\{k \in B2 \text{ t.q } k < i\} + i < 38$. Esto se debe a que $B2^{'} = B2$ y a que i1<i. Y por tanto la distribución resultante de hacer un cambio más fuerte es también no válida.

ii) Si el cambio (i,j) es no válido por cumplir ii) siguiendo un razonamiento análogo se sigue que la distribución resultante de hacer un cambio más fuerte que este es también no válida.

iii) En el caso de que el cambio (i,j) sea no válido por cumplir iii) implica que:
$\{k \in B2 \text{ t.q } k < i\} = \emptyset$. Luego si se hace el cambio más fuerte (i,j1) con j1>j lo anterior se seguirá cumpliendo para $B2^{'}$ (recordar j,j1>i). Si en cambio se hace el otro cambio más fuerte (i1,j) con i1<i dicho conjunto seguirá siendo vacio puesto que: $\{k \in B2^{'} \text{ t.q } k < i1\} \subset \{k \in B2 \text{ t.q } k < i\} = \emptyset$, como antes $B2^{'} = B2$ e i1<i. Luego la distribución resultante del cambio más fuerte será no válida.

iv)En el caso de que el cambio (i,j) sea no válido por cumplir iv) con un razonamiento análogo al anterior se ve que la distribución resultante sigue siendo no válida.

Teorema 3. *Si en un cambio de n equipos sobre cierta distribución básica hay al menos un cambio de 1 equipo no válido, la distribución resultante del cambio de n equipos sobre dicha distribución básica será no válido.*

Demostración: Se verá este cambio de n equipos sobre una cierta distribución básica como un cambio 1 no válido sobre dicha distribución seguido de un cambio de n-1 equipos sobre la distribución resultante. Por ello cuando hable de valor anterior me referiré al valor que habrá en cierta característica de la distribución intermedia antes de hacer los n-1 cambios restantes. Llamaré (B2,B3) a la distribución intermedia (resultante del cambio 1 sobre la distribución básica) y $(B2^{'}, B3^{'})$ a la distribución resultante de hacer los n cambios sobre dicha distribución básica.
Si el cambio (i,j) es no válido la distribución intermedia (B2,B3) cumplirá una o más de las siguientes condiciones de la definición 5:

i)$8 + max\{j \in B2 \text{ t.q } j < min\{k \in B3\}\} + min\{k \in B3\} < 38$. Es claro que al aplicarle un cambio de n-1 equipos (que no involucra a los equipos del cambio 1) a la distribución intermedia, el nuevo $min\{k \in B3^{'}\}$ será menor o igual que el anterior puesto que por la definición de cambio de k equipos sobre una distribución los nuevos elementos de $B3^{'}$ con respecto a B3 serán menores que los que se van. Y por la misma razón el nuevo $max\{j \in B2^{'} \text{ t.q } j < min\{k \in B3^{'}\}\}$ será menor o igual que el anterior. Luego se seguirá cumpliendo i), luego este cambio de n-1 equipos seguirá dando lugar a una distribución no válida, es decir el cambio de n equipos será no válido.

ii) Se sigue de manera análoga a i), sigue dando una distribución no válida.

iii) Al igual que en i) el $min\{k \in B3^{'}\}$ será menor o igual que el anterior y los nuevos elementos de $B2^{'}$ con respecto a B2 son mayores que dicho mínimo (definición cambio k equipos y distribución básica) luego: $\{j \in B2^{'} \text{ t.q } j < min\{k \in B3^{'}\}\} \subset \{j \in B2 \text{ t.q } j < min\{k \in B3\}\} = \emptyset$. Luego esta nueva distribución resultante de hacer n cambios es no válida.

iv)Se sigue de manera análoga a iii).

En nuestro caso no habrá cambios de 2,3,4,5,6,7,8 equipos, ya que todas las distribuciones resultantes de este tipo de cambios serán no válidas. Esto se debe a que para cada distribución básica los únicos cambios válidos (generan una distribución que no se puede definir como no válida) son (16,17) y (16,18). Luego cualquier cambio de n equipos (con n>1) tendrá al menos un cambio 1 no válido. Esto es claro por el hecho de que solo hay dos cambios 1 válidos y no son disjuntos. Así las distribuciones que generarán todas las soluciones son:
i)B4=B4A: Sorteo básico, cambio (16,17), cambio (16,18).
ii)B4=B4B: Sorteo básico, cambio (16,17), cambio (16,18).
iii)B4=B4C: Sorteo básico, cambio (16,17), cambio (16,18).
iv)B4=B4D: Sorteo básico, cambio (16,17), cambio (16,18).
v)B4=B4E: Sorteo básico, cambio (16,17), cambio (16,18).
Cambio (16,17) o (16,18) se refiere a ese cambio sobre la distribución básica correspondiente.
Una vez conocidas las distribuciones que generaran todas las soluciones, usando el programa Free-Pascal creo una serie de programas que tardan menos de 1 minuto en dar todas las soluciones. En el anexo pondré un ejemplo de estos programas (son varios pero todos con la misma estructura, solo varían algunos números). Por tanto el algoritmo descrito me ha permitido pasar de un programa que tardaría del rango de 4.000 días a un serie de pequeños programas que tardarán menos de 1 minuto en darnos las soluciones.

Capítulo 3

Creación de los sorteos.

Haciendo los programas necesarios siguiendo el algoritmo explicado en la parte anterior hallo todos las agrupaciones cuyos grupos suman 38, es decir, todas las formas de repartir los 32 equipos en 8 grupos de forma que los 3 equipos con menos peso de cada grupo sumen 38. Respetando las restricciones geográficas impuestas por la FIFA, y el hecho de que los 8 cabezas de serie no pueden ir en el mismo grupo siendo el cabeza de serie el anfitrión de la Copa del Mundo.
La lista de todas las agrupaciones perfectas se compone de 173.160 agrupaciones. Pasaré ahora a proponer 2 maneras en las que se podría encarar este sorteo y que por definición de sorteo bueno, habrán de ser sorteos equiprobables. Habría muchas maneras diferentes de crear estos sorteos, las que voy a explicar me parecen las más representativas.

1-Una vez halladas todas las posibles soluciones, se haría una lista con ellas de 173.160 elementos (y se numerarían del 0-173.159). Este primer sorteo trataría de tener 6 bombos: el primero con una bola que contenga el número 0 y otra el 1. Y los otros 5 bombos contendrían bolas con los números del 0-9. De este modo se cogería una bola de cada bombo y se obtendría un número que iría asociado a una solución y esa sería la que se usaría, es decir, ese sería el resultado del sorteo.
Notar que para que el sorteo sea equiprobable lo que no se puede hacer sería que en el caso de que saliera un 1 en el primer bombo quitar las bolas 8,y 9 del 2^{o} bombo ya que eso impediría que el sorteo fuera equiprobable. En consecuencia podría darse el caso de que se obtuviera un número fuera del rango de soluciones, en dicha situación se reharía el sorteo. La probabilidad de que el primer número sea bueno es: $p = \dfrac{\text{Números aceptables}}{\text{Números totales}} = \dfrac{173{,}160}{200{,}000} = 0{,}8658$. Es decir la probabilidad de que se necesite hacer 2 veces dicho sorteo o menos es de : $p + (1-p)p \simeq 0{,}9812$, con lo que sería un sorteo viable para hacer de cara al público (si se necesitaran de 5 o más sorteos, no quedaría bien de cara a este).

2-Siguiendo la idea del sorteo anterior, una pequeña modificación sería la de escoger unas 10 soluciones (fuera de cámaras por asi decirlo) y luego de cara al público presentar las 10 soluciones finalistas y hacer un sorteo de una de estas. La parte negativa de esta opción sería que podría darse el caso de que salieran soluciones tremendamente parecidas (ya que hay muchas soluciones que solo difieren de otras en que se han intercambiado el cuarto equipo entre 2 grupos), para evitar esto sería recomendable sacar más soluciones en caso de que ocurriera esto (pero no desechar ningún porque sea repetitiva, se perdería la equiprobabilidad). La cantidad de soluciones que se quieran sacar y el

cómo descartarlas tiene muchas maneras de hacerse (con cuidado de no perder la equiprobabilidad), en mi opinión coger 10, 16 o 20 sería una buena opción.
Desde mi punto de vista la segunda forma de sorteo explicada sería la más visual y la que mejor aceptación tendría de las 2 expuestas.

3.1. HEURÍSTICA DEL SORTEO 323

Tras crear el sorteo perfecto, se ve que este tiene un pequeño fallo (si es que se le puede llamar así), que es que de cara al público o de cara a hacer un show de esto, no es muy atractivo. Por lo tanto el siguiente objetivo es el de crear un segundo sorteo que no solo ha de ser llamativo de cara al público, sino que además ha de mejorar al sorteo propuesto por Julien Guyon (como ya se ha explicado cuando se habla de mejorar me refiero a que la suma de los pesos de los tres equipos más fuertes de cada grupo sea lo más igualada posible) intentando no empeorar en exceso la suma total del grupo. El sorteo perfecto era óptimo en este aspecto, pero no es del todo atractivo de cara al público, por tanto rebajaré un poco las pretensiones de perfección, pero no tanto como para que el de Guyon lo supere en este aspecto.

Antes de empezar a pensar en como hacer dicho sorteo, hay que ver si hay algún punto débil en el sorteo de Guyon. Antes de empezar con esto hay que dejar tres puntos claros:
1- Aunque lo que se quiere es mejorar la suma de los tres primeros, habrá que incluir en algún momento al cuarto equipo, de una manera razonable e intentando que no dañe en exceso la igualdad de la suma total de cada grupo.
2- Como se quiere hacer un sorteo lo más genérico posible, es decir, no puede ser solo aplicable al caso de la Copa del Mundo 2014, se ignorarán momentáneamente las restricciones geográficas, a la hora de crearlo, que no a la hora de ejecutarlo.
3- Se seguirán usando las hipótesis de la FIFA de que los 8 cabezas de serie no pueden coincidir en el mismo grupo.

Teniendo en mente la forma de crear el sorteo de Guyon (ver Introducción), el primer problema que se aprecia (omitiré momentáneamente en los siguientes razonamientos al cuarto bombo) es el siguiente: en media cada grupo recibirá 37.5 puntos de peso, pero al separar el sorteo de Guyon en dos partes en una parte su media será 35.5 y en la otra 39.5 lo que asegura que habrá un grupo de peso 35 o menos y otro de peso 40 o más como mínimo (es decir el rango en el mejor de los casos es 5). Este aspecto no es la verdadera raíz del problema, pues reordenando el bombo 3 se puede conseguir fácilmente que la media sea 37.5 en cada sorteo. El problema es que independientemente de como se ordenen el bombo 1, 2 y 3 (sin olvidar las hipótesis FIFA) habrá casos malos, llamando caso malo a un resultado de un sorteo en el que alguno de los grupos tiene un peso demasiado pequeño o demasiado grande (recordar que el cuarto bombo por el momento no existe). Esto no quiere decir que el sorteo de Guyon sea malo sino que es mejorable y la forma de mejorarlo sería evitar la existencia de esos casos. Por tanto la siguiente cuestión a resolver es: ¿Cómo se producen esos casos en el sorteo de Guyon?¿Y cómo resolver este problema?

Lo primero que haré será tratar a los equipos de peso 25-32 cómo los equipos más pesados de cada grupo, es decir, formarán el bombo 4 con el que trabajaré al final. Por lo tanto la media de

peso de los 3 equipos más fuertes de cada grupo será 37.5, por ello lo ideal sería que todos tuvieran peso 37 o 38 y se podría considerar un mal resultado si algún grupo tuviera 33 o menos o 42 o más (aproximadamente). Es decir, se quiere evitar en la medida de lo posible que se den esos casos. Un par de ejemplos de esos casos serían: 2-13-17, 8-12-23. Este tipo de casos no son muy comunes pero la existencia de estos implica la posibilidad de encontrar grupos malos. En un análisis más detallado de los bombos de Guyon se puede ver que cada bombo se puede dividir en tres partes, parte alta, parte media y parte baja; Ejemplo con bombo 1:
(1,2):parte alta, (3,4,5,6):parte media, (7,8):parte baja).
E indagando un poco más se ve que la mayoría de los casos malos se producen cuando dos equipos de la parte alta de su respectivo bombo, o de la baja coinciden en el mismo grupo. Es decir si se consigue evitar que las partes altas coincidan con las mismas en el grupo y lo mismo para el caso de las partes bajas, se habrá quitado una gran parte de los casos malos.
Pero hay un problema, por mucho que se reordenen (sin intercambiar equipos entre bombos) los bombos 1,2 y 3 de Guyon, hay tres partes altas y tres partes bajas que han de ser asignadas a dos sorteos; se haga lo que se haga en uno de los sorteos como mínimo coincidirán dos partes del mismo tipo, que es lo que se quiere evitar.
La solución con la que se solventará este problema es la que también da nombre al sorteo, el 323. Lo que se hace es dividir cada bombo en tres partes, luego se pasará de tener dos sorteos independientes a tres, de modo que en cada parte haya tres pesos consecutivos (dos en el caso de 2). Pondré un ejemplo de esta subdivisión de bombos con un cuadro:

1	2	3	4	5	6	7	8
16	15	14	13	12	11	10	9
17	18	19	20	21	22	23	24

Cuadro 3.1: Ejemplo de distribución 323.

Los números en rojo formarían uno de los sorteos independientes, el otro sería el azul y el otro el negro. Como se puede ver en ninguno de los sorteos coexisten dos partes altas o dos partes bajas, es decir, se ha quitado gran parte de los casos malos. Compararé el rango de valores que puede tomar un grupo en cada sorteo, refiriéndome a la suma de los 3 más fuertes:
Guyon; Parte alta: 31-40. Parte baja: 35-44.
323; Rojo: 35-41. Azul: 36-39. Negro: 34-40.
La diferencia es clara (más adelante se analizarán estos resultados con más profundidad), luego parece que este tipo de sorteo va bien encaminado a la hora de mejorar el de Guyon.
Esta será la estructura que seguirá el sorteo, procederé ahora a explicar como se haría en el caso de la Copa del Mundo 2014 (intentando ser lo más genérico posibles):

Paso 1: Con la estructura del 323 hay seis posibles modos de dividir cada bombo de la forma expuesta y de que en cada sorteo haya una parte alta una baja y una media (la alta y baja contienen cada una a los elementos definidos como pertenecientes a ésta). A mi parecer la mejor (de cara al público) es en la que se separa el bombo 1 en 323 (y no en 233 o 332), una vez fijado este bombo el bombo 2 y bombo 3 se pueden reordenar de dos maneras, una es la expuesta en el cuadro 2 y la otra sería: (1-2-3; 12-13-14; 22-23-24 , 4-5;15-16;17-18 y 6-7-8;9-10-11;19-20-21). Estas serán las estructuras con las que se creará sorteo.

Paso 2: Análisis de las estructuras: Si ahora se tiene en cuenta las restricciones geográficas, se ve que la segunda manera es inviable (en la parte del 6-7-8;9-10-11;19-20-21 habrá seguro un grupo con tres europeos al haber 7 de estos). Y en la primera por cuestiones geográficas el equipo de peso 20 siempre irá con el equipo de peso 1. Como el objetivo es que este sorteo sea más atractivo de cara al público no se puede permitir que sin ningún sorteo esos dos equipos tengan que ir en el mismo grupo. Por lo que a costa de perder un poco de calidad en el sorteo (a nivel estadístico), se harán unas pequeñas modificaciones que intentaré sean lo menos agresivas posibles. La modificación será la siguiente: se crearán nuevos sorteos, partiendo de una de las 2 estructuras iniciales se intercambiará un solo equipo con otro de dos sorteos diferentes que compartan el bombo del que son una parte. De manera que en cada subbombo (partes en que ha sido dividido cada bombo) la diferencia entre el equipo más pesado y el menos pesado sea como mucho de 3. Es decir: el cambio 13-14 es viable pero no así el 9-16 ni el 16-17 (no se mezclarán los bombos iniciales). Esta parte es opcional para otros mundiales, en este caso se ha impuesto para aumentar la variedad de resultados.

Paso 3: Una vez hallados todos los nuevos sorteos posibles, añadidos al inicial que tenía, se descartarán los que sean inviables por restricciones geográficas. Una vez hecho esto se procederá a introducir al cuarto bombo en el sorteo. Para ello se dividira el cuarto bombo como antes en tres partes (hay tres formas posibles) y se calculará mediante un programa informático a que sorteo debe ir cada subbombo 4 (hay seis formas diferentes), para lo cual calcularé mediante dicho programa cuál de las seis distribuciones da una media de la varianza de 4 menor (sin tener en cuenta las restricciones geográficas a la hora de calcularlas) y esa distribución será la que completará el sorteo. En el caso de que la distribución óptima para alguna de los sorteos anteriores cree un sorteo inviable, dicho sorteo se eliminará. Y en caso de que halla dos distribuciones con el mismo resultado se elegirá la que de más variedad al sorteo global (es decir, si hay muchos sorteos en los que la estructura A es la mejor y en uno hay un empate entre la estructura A y B se elegirá la B; en caso de empate en este aspecto se sorteará cuál usar). El objetivo con esto es no dañar en exceso el peso total del grupo, es decir, aunque el objetivo principal es mejorar la suma de los tres primeros no interesa destrozar la suma total.

Paso 4: En caso de que no se halla logrado encontrar ningún sorteo válido de la forma en que lo he propuesto, se procedería a crearlos de la siguiente manera: En vez de trabajar con la distribución del bombo 1 323 se trabajaría con la 332 y se haría lo ya explicado (incluidos los cambios si fueran necesarios y el crear los subbombos 4 de manera óptima). Si esto tampoco funcionará se haría lo mismo con el 233. Y en el caso de que esto tampoco diera resultado (algo que sería bastante difícil) se procedería a relajar algunas condiciones, es decir, se podrían hacer varios cambios a la vez y/o no exigir que la diferencia entre pesos sea como mucho 3 y como última opción elegir la forma de distribuir los subbombos 4 de forma que puedan crear sorteo viables ignorando la optimalidad.

Con esto ya se puede crear el sorteo en cualquier tipo de situación. En el caso del mundial 2014 se obtienen 6 sorteos diferentes. Procederé ahora a explicar como se haría el sorteo:

Paso 1:Como en cada sorteo no hay el mismo número de soluciones posibles (y el sorteo ha de ser equiprobable), se hará un primer sorteo de sorteos para ver cuál se usa. Y la probabilidad de que salga un sorteo u otro dependerá de la cantidad de casos que tenga cada sorteo.

Paso 2:Una vez elegido el sorteo ha realizar, se hará un procedimiento similar al de Guyon; se sorteará la distribución geográfica del sorteo que haya salido en el Paso 1.

Paso 3:Por último se sorteará a los equipo de acuerdo a su continente y bombo.

Este sorteo será equiprobable (demostración análoga a la hecha para el sorteo de Guyon en su anexo).

Capítulo 4

Análisis de resultados.

Tras haber presentado los 4 tipos diferentes de sorteos con los que trabajaré: 323, Guyon, FIFA y Perfecto, procederé ahora a compararlos a nivel estadístico usando el programa RStudio. Para ello se tomará como referencia las medidas de dispersión: rango 3, rango 4, varianza 3 y varianza 4:
Llamando (i_m, j_m, k_m, l_m) t.q $i_m < j_m < k_m < l_m$ a los equipos que forman el grupo m $(m = 1, ..., 8)$, estas medidas vienen definidas como:

$Rango3 = \max_{m=1,...,8}\{i_m + j_m + k_m\}$

$Rango4 = \max_{m=1,...,8}\{i_m + j_m + k_m + l_m\}$

Notación: $\mu = \sum_{m=1}^{8} i_m + j_m + k_m/8$

$Varianza3 = \sum_{m=1}^{8} (i_m + j_m + k_m - \mu)^2/8$

$Varianza4 = \sum_{m=1}^{8} (i_m + j_m + k_m + l_m - 66)^2/8$

Para hacer este estudio se realizarán 10.000 sorteos de los tipos: 323, Guyon y FIFA. Y se usará la población del perfecto, esto se debe a que se tiene una enumeración de todos sus resultados.
A continuación se hará un resumen numérico de los resultados obtenidos en los 10.000 sorteos de cada tipo, para luego proceder a comparar las medias del rango y la varianza (se especificará más adelante entre qué sorteos y porqué) y luego se harán unas gráficas para que de una forma más visual se vean las diferencias existentes entre los sorteos.

RANGO 3:

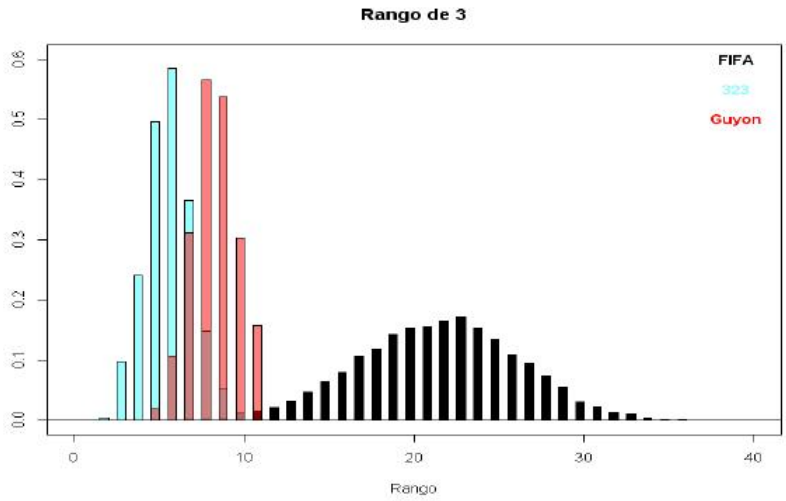
	mean	sd	IQR	0%	10%	25%	50%	75%	90%	100%
Rgo3.323	5.7937	1.4022	2	2	4	5	6	7	8	10
Rgo3.FIFA	21.6141	4.7237	7	6	15	18	22	25	28	37
Rgo3.guyon	8.5175	1.3285	1	5	7	8	8	9	10	11

Cuadro 4.1: Resumen numérico del rango de 3 de las 10.000 muestras. Notación: mean: Media; sd: desviación estándar ; IQR: rango intercuartílico; resto: percentiles

A la vista de los resultados solo merece la pena comprobar si la media del rango 3 coincide en el sorteo de guyon y en el 323, puesto que por el resumen de los datos del sorteo FIFA, es bastante claro que sus medias no coincidirán. Por la misma razón no se va a mirar si estas mismas medias son 0(es

decir si coinciden con la del sorteo perfecto).
Para comparar las medias del 323 con el sorteo Guyon, se aplicará un test de hipótesis sobre el supuesto de que coinciden y por el tamaño de las muestras se usará el test de comparación de medias para muestras independientes de la t de student.
Hipotesis nula:las medias son iguales;Guyon-323:
t = -141.0064, df = 19939.99, p-value < 2.2e-16
95 percent confidence interval: -2.761663 -2.685937

Como se puede observar por el resultado del test las medias no coinciden (el p-valor es muy pequeño y por tanto se rechaza la hipótesis nula de que las medias eran iguales). Procederé ahora a poner un gráfico de la distribución de los valores de las muestras:



Gráfica 4.1: Distribución de los valores de la muestra.

A la vista de los resultados se ve lo que ya se podía intuir con el resumen numérico, el sorteo 323 es considerablemente mejor en este aspecto que el de Guyon; y el de la FIFA es pésimo en este aspecto. Notar que aunque no se haya puesto en el gráfico el sorteo Perfecto tiene $rango3 = 0$.

RANGO 4:

	mean	sd	IQR	0%	10%	25%	50%	75%	90%	100%
Rgo4.323	7.4101	2.0566	3	2	5	6	7	9	10	14
Rgo4.FIFA	28.7829	7.4105	10	6	19	24	29	34	38	54
Rgo4.guy	5.7878	1.7911	2	1	4	5	6	7	8	11
Rgo4.perf	9.4464	0.6598	1	8	9	9	9	10	10	11

Cuadro 4.2: Resumen numérico del rango 4 de las 10.000 muestras y de la población del Perfecto. Misma notación que en el Cuadro 4.1.

Al igual que antes procederé a comparar las medias del sorteo 323, Guyon y Perfecto respectivamente, por la misma razón que antes no se meterá en esta comparación al sorteo FIFA. Al igual que antes se harán unos test de hipótesis:

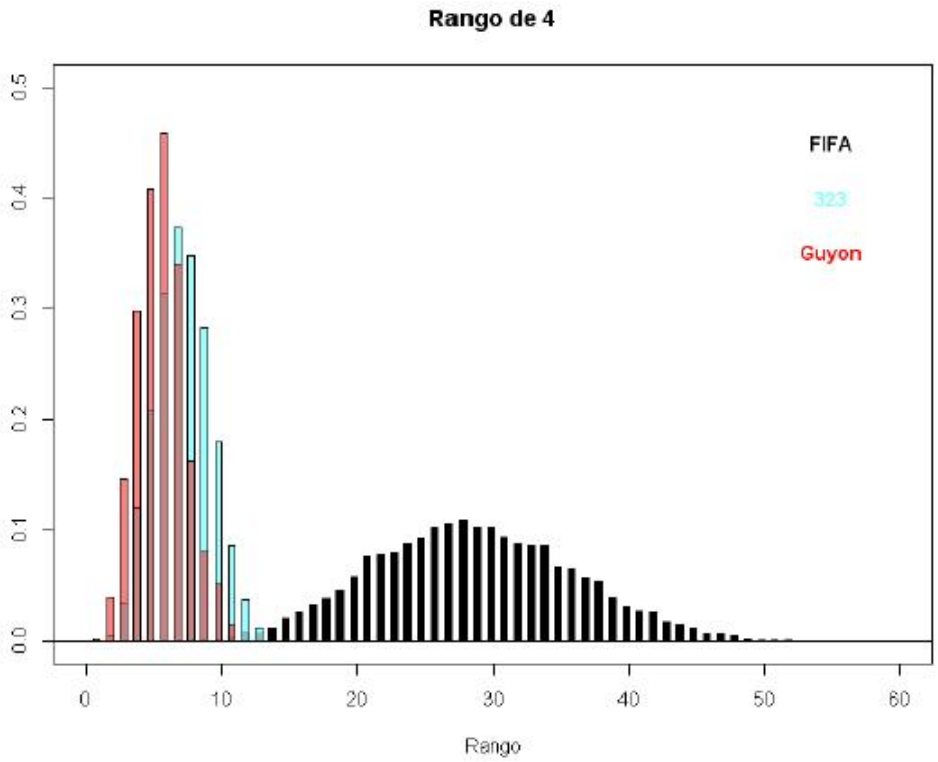
Hipotesis nula:las medias son iguales;Guyon-323:
t = 59.4835, df = 19627.72, p-value < 2.2e-16
95 percent confidence interval: 1.568842 1.675758

Hipotesis nula:la media del 323 es igual a la media de la población del Perfecto:
t = -99.009, df = 9999, p-value < 2.2e-16
95 percent confidence interval: 7.369785 7.450415

Hipotesis nula:la media de Guyon es igual a la media de la población del Perfecto:
t = -204.2575, df = 9999, p-value < 2.2e-16
95 percent confidence interval: 5.752689 5.822911

En el caso de comparación de medias Guyon-323 se ha usado el mismo test que en la parte anterior. En cambio para la comparación de medias del 323 y Guyon con la media de la población del sorteo Perfecto, debido al tamaño de la muestra y al conocimiento de la media de la población del Perfecto, se ha aplicado el test t de comparación de medias de una muestra de la t de Student.

Luego a la vista de los resultados queda rechazada la hipótesis de que las medias de alguno de los sorteos sean iguales (por el hecho de que los p-valores son demasiado pequeños). Pasaré a hacer un gráfico para ver de forma más visual la distribución de las muestras:



Gráfica 4.2: Distribución de los valores de la muestra.

Al igual que antes se aprecia una enorme diferencia entre el sorteo FIFA y el de Guyon y 323, pero en este caso es el de Guyon el que tiene una ligera ventaja sobre el 323 (notar que esta ventaja se podría

considerar menor que la existente en el rango 3).

VARIANZA 3:

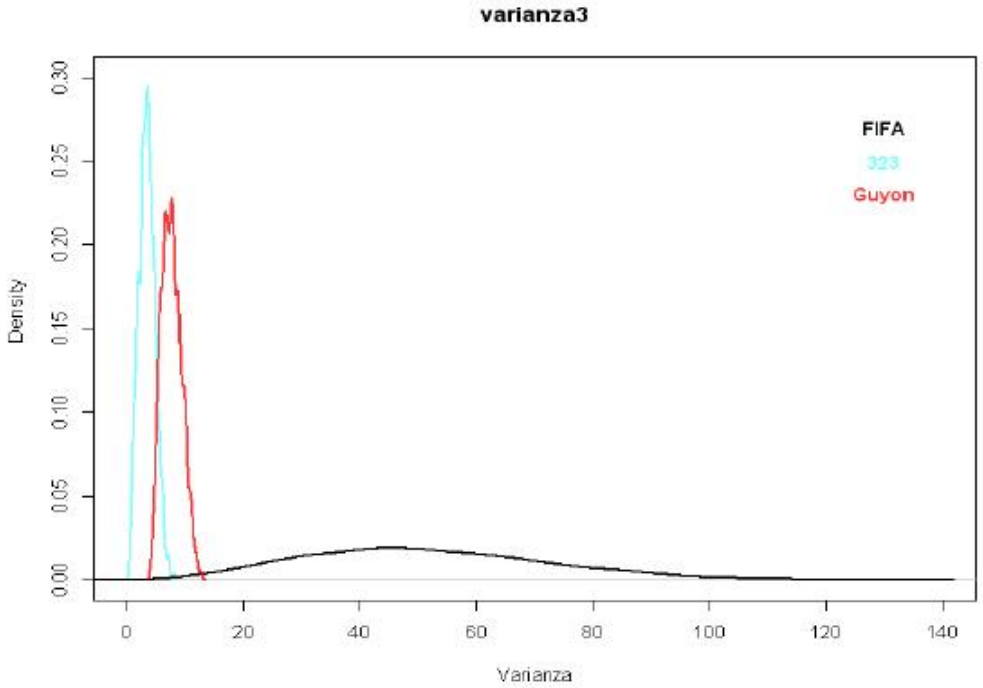
	mean	sd	IQR	0%	10%	25%	50%	75%	90%	100%
Var3.de.323	3.5604	1.3706	1.75	0.50	1.75	2.75	3.50	4.50	5.25	8.25
Var.3.FIFA	51.0921	20.7364	28.98	2.98	25.25	35.75	49.36	64.73	79.23	133.00
Var.3.guyon	7.7630	1.6780	2.50	4.25	5.75	6.50	7.75	9.00	10.0	12.75

Cuadro 4.3: Resumen numérico de la varianza 3 de las 10.000 muestras. Misma notación que la usada en Cuadro 4.1.

Al igual que en casos anteriores se procederá únicamente a comparar la media de dichas varianzas de los sorteos 323 y Guyon usando para ello el test para muestras independientes antes mencionado: Hipótesis nula: Las medias de los sorteos Guyon-323 son iguales:

t = -193.9676, df = 19231.7, p-value < 2.2e-16
95 percent confidence interval: -4.245068 -4.160132

Como en casos anteriores al ser el p-valor pequeño se rechaza la hipótesis, es decir las medias no coinciden. Procederé ahora a hacer un gráfica para ver de manera más visual las diferencias existentes:



Gráfica 4.3: Distribución de los valores de la muestra.

Como se puede observar hay una diferencia considerable entre el sorteo 323 y el sorteo de Guyon, y como en casos anteriores el sorteo de la FIFA presenta unos resultados muy malos en comparación con los otros sorteos. Notar que aunque no se haya puesto en el gráfico el sorteo perfecto tiene varianza 3 = 0.

VARIANZA 4:

	mean	sd	IQR	0%	10%	25%	50%	75%	90%	100%
Var.4.de.323	5.5955	2.4077	3.50	0.50	2.25	3.75	5.5	7.25	8.75	13.75
Var.4.FIFA	86.7791	38.4219	53.50	4.50	39.50	58.00	83.0	111.50	139.25	247.00
Var.4.guyon	4.9001	2.3514	3.25	0.25	2.00	3.25	4.5	6.50	8.00	14.75
Var.4.perf	9.0708	0.6090	1.00	7.50	8.50	8.50	9.0	9.50	9.50	10.50

Cuadro 4.4: Distribución de los valores de la muestra. Misma notación que en el Cuadro 4.1.

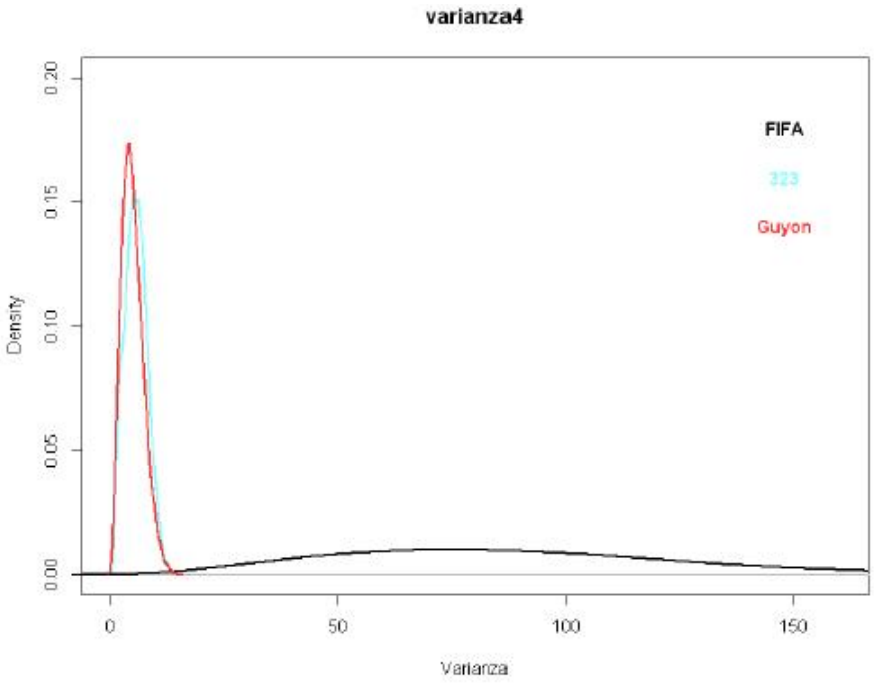
Siguiendo el razonamiento de los casos anteriores compararé la media de la varianza 4 entre el sorteo de Guyon, el 323 y el valor obtenido de la población del sorteo perfecto. Se aplicarán los mismos test que los aplicados para el rango 4:

Hipótesis nula: Las medias del sorteo 323 y Guyon son iguales:
t = 20.6618, df = 19986.83, p-value < 2.2e-16
95 percent confidence interval: 0.6294084 0.7613416

Hipótesis nula : La media del 323 es igual a la media de la población del sorteo perfecto:
t = -144.339, df = 9999, p-value < 2.2e-16
95 percent confidence interval: 5.548304 5.642696

Hipótesis nula : La media de Guyon es igual a la media de la población del sorteo perfecto:
t = -177.3644, df = 9999, p-value < 2.2e-16
95 percent confidence interval: 4.854031 4.946219

Quedan desechada las hipótesis de que las medias de la varianza 4 sean iguales entre estos sorteos debido a que le p-valor es muy pequeño. Como antes pasaré a ver este hecho desde un punto de vista más visual usando una gráfica de la distribución de los valores de la muestra.



Gráfica 4.4: Distribución de los valores de la muestra.

En este caso se puede ver cómo el sorteo de Guyon obtiene mejores resultados que el 323, pero la diferencia es casi inapreciable. El sorteo FIFA sigue dando pésimos resultados.

Capítulo 5

Conclusiones.

A la vista de los resultados obtenidos se llegan a varias conclusiones:
a) El sorteo FIFA es un pésimo sorteo si se parte de la base de que para que un sorteo sea bueno los grupos han de tener un potencial semejante.
b) En cuanto a cuál es mejor de los otros sorteos, depende de la forma en que se quiera enfocar el sorteo. Si lo que se quiere es la mayor igualdad posible, el sorteo perfecto es el mejor (puesto que en este todos los grupos tienen el mismo potencial, teniendo en cuenta a los tres mas fuertes de cada grupo). Si en cambio también se quiere tener en cuenta a todos los integrantes del grupo, inicialmente el sorteo de Guyon es el mejor, pero tras ver los resultados estadísticos sería mejor el del 323 puesto que a pesar de empeorar ligeramente cuando se tiene en cuenta a todos los del grupo, la mejora es sustancial cuando solo se tiene en cuenta a los tres más fuertes.
c) Luego mi propuesta sería la de usar el sorteo perfecto (puesto como ya he explicado me parece más importante el hecho de que el peso de los 3 más fuertes sea muy similar en cada grupo que el que lo sea el peso de todos los integrantes de grupo) como primera opción, y en caso de que no fuera aceptada por problemas de implementación o por ser poco atractivo de cara al público, optaría por el sorteo 323 frente al sorteo Guyon por lo ya explicado anteriormente: el 323 es bastante mejor sobre el peso de los 3 más fuertes y el de guyon sensiblemente mejor teniendo en cuenta la totalidad del grupo.
d) Por último notar que los 2 sorteos que se han creado en este texto pueden ser aplicados independientemente del ranking que se use.

5.1. Ampliación

a) En este trabajo el método 323 se ha enfocado únicamente a la mejora de la suma de los 3 primeros, pero siguiendo la misma estructura de sorteo, pero enfocado a la totalidad del grupo, se podría crear un sorteo que mejorara al de Guyon en cuanto a la suma de los 4 del grupo.
b) También se podría enfocar el sorteo perfecto a la búsqueda de grupos en los cuales todos tengan el mismo potencial (teniendo en cuanta a los 4). Aunque el hallar todos los grupos perfectos de este tipo sería más difícil que antes, ya que si se siguiera un procedimiento similar al ya expuesto la cantidad de distribuciones diferentes a programar sería muy grande y los programas tardarían bastante en compilarse ya que no se podrían dividir en 2 programas como si se podía en nuestro caso.
c)Por último los sorteos creados en este trabajo se podrían aplicar al sorteo de la UEFA Champions League que sigue la misma estructura que el del Mundial (32 equipos repartidos en 8 grupos), y a cualquier otro sorteo de la misma estructura que mantenga la restricción de los cabezas de serie.

Bibliografía

[1] Aisch, G., & Leonhardt,D. (2014). Mexico, the World's Cup luckiest country. *New York Times.*

[2] Bucholtz,A. (2013). World Cup 'Potgate' scandal sees France earn easy draw, Italiy in though, cries of conspiracy. *Eh Game blog post.*

[3] Guyon, J. (2014). Rethinking the FIFA World Cup^{TM} Final Draw. *Journal on Quantitative Analysis in Sports.*

[4] Jones, M.C.(1990). The World Cup draw's flaws. *The Mathematical Gazette, 335-338.*

[5] Klößner, S., & Becker, M.(2013). Odd Odds: The UEFA Champios League Round of Sixteen Draw. *Journal of Quantitative Analysis in Sports, 9(3), 249-270.*

[6] Prodinger, H. (1982). Number of partitions of $(1,...,n)$ into two subsets of equal cardinalities and equal sums. *CAN. MATH. BULL., 25(2), 238-241.*

[7] Prodinger, H. (1984). On the number of partitions of $\{1,...,n\}$ into r sets of equal cardinalities and sums. *Tamkang Journal of Mathematics, 15, 161-164.*

[8] Taha, H.A.(1975). Integer programming: theory, applications and computations. *Academic Press, New York.*

[9] página web FIFA, *Sorteo FIFA World Cup 2014, Brasil*, disponible en: *http://www.fifa.com/mm/document/tournament/finaldraw/02/23/84/73/131203 _ finaldrawprocedures _ neutral.pdf.*

[10] página web FIFA, *Ranking FIFA usado en el sorteo, Octubre 2013*, disponible en: *http://www.fifa.com/fifa-world-ranking/ranking-table/men/rank=231/index.html*

[11] pagina web FIFA, *Ranking FIFA, método de aplicación*, disponible en: *http://es.fifa.com/fifa-world-ranking/procedure/men.html*

[12] Programa FreePascal, disponible en: *http://free-pascal.softonic.com/*

[13] Manual FreePascal, disponible en: *http://www.freepascal.org/docs-html/*

[14] Código errores FreePascal, disponible en: *https://there10han.wordpress.com/programming/pascal/free-pascal-exitcoderuntime-error-code/*

[15] Programa R studio, disponible en: *r-studio.softonic.com/*

Anexo A

Programación dinámica

A.1. Búsqueda de los posibles B4.

Procederé ahora a calcular todos los posibles B4, o lo que es lo mismo a hallar:
$(x_1,x_2,x_3,x_4,x_5,x_6,x_7,x_8)$ t.q $x_1+x_2+x_3+x_4+x_5+x_6+x_7+x_8=224$,
$x_1<x_2<x_3<x_4<x_5<x_6<x_7<x_8$ con $x_i \in \{9,10,...,31,32\}$ para $i=1,...,8$.

Para ello primero probaré que algunos de estos x_i solo pueden tomar un valor, y de esta forma se simplificará el problema antes de resolverlo mediante programación dinámica:

1) $x_8=32$. Suponer que $x_8 \neq 32$ la máxima suma que se puede conseguir sería: $24+25+26+27+28+29+30+31=220<224$. Luego es imposible que $x_8 \neq 32$.
b) Nuevo problema: $x_1+x_2+x_3+x_4+x_5+x_6+x_7=192$ t.q $x_1<x_2<x_3<x_4<x_5<x_6<x_7$ con $x_i \in \{9,10,...,31\}$ para $i=1,...,7$.

2) $x_7=31$. Suponer que $x_7 \neq 31$ la máxima suma que se puede conseguir sería: $24+25+26+27+28+29+30=189<192$. Luego es imposible que $x_7 \neq 31$.
c) Nuevo problema: $x_1+x_2+x_3+x_4+x_5+x_6=161$ t.q $x_1<x_2<x_3<x_4<x_5<x_6$ con $x_i \in \{9,10,...,30\}$ para $i=1,...,6$.

3) $x_6=30$. Suponer que $x_6 \neq 30$ la máxima suma que se puede conseguir sería: $24+25+26+27+28+29=159<161$. Luego es imposible que $x_6 \neq 30$.
d) Nuevo problema: $x_1+x_2+x_3+x_4+x_5=131$ t.q $x_1<x_2<x_3<x_4<x_5$ con $x_i \in \{9,10,...,29\}$ para $i=1,...,5$.

4) $x_5=29$. Suponer que $x_5 \neq 29$ la máxima suma que se puede conseguir sería: $24+25+26+27+28=130<131$. Luego es imposible que $x_5 \neq 29$.
e) Nuevo problema: $x_1+x_2+x_3+x_4=102$ t.q $x_1<x_2<x_3<x_4$ con $x_i \in \{9,10,...,28\}$ para $i=1,...,4$.

5) $x_1>20$. Suponer que $x_1=20$. La máxima suma que se puede conseguir sería: $20+26+27+28=101<102$ luego queda probado que $x_i \in \{21,...,28\}$ para $i=1,...,4$ (debido a que los x_i son crecientes).

Es decir, lo que se ha averiguado es que:
$x_8 = 32, x_7 = 31, x_6 = 30, x_5 = 29$ y que $x_1, x_2, x_3, x_4 \in \{21, ..., 28\}$ t.q $x_1 + x_2 + x_3 + x_4 = 102$ con $x_1 < x_2 < x_3 < x_4$.

Procederé a enunciar este problema como un problema de programación dinámica para resolverlo de este modo:
Hay 4 etapas (una por cada x_i a decidir). Y la función objetivo que se quiere minimizar es una que diga si se han ido tomando las decisiones correctas.

Se denotará: $X_k = (x_k, \sum_{i=1}^{k-1} x_i)$. Dicho sumatorio será 0 en el caso de que $k = 1$. Denotaré ademas a $z_k = \sum_{i=1}^{k-1} x_i$
Sea: $g_k(X_k, d_k) = \mid z_{k+1} - x_k - z_k \mid$ el valor que indica si se ha tomado la decisión correcta, si es así tendrá valor 0. Y $g_4(X_4) = 0 = J_4(X_4)$. El objetivo será : $\min_{d_1,...,d_3} \{\sum_{k=1}^{4} g_k(X_k, d_k) + g_4(X_4)\}$. Siendo $d_k = X_{k+1}$ y $J_k(X_k) = \min_{d_k}\{g_k(X_k, d_k) + J_{k+1}(X_{k+1})\}$, que si es 0 indicará que se está tomando la solución correcta.

X_4 \ d_4	102	$J_4(X_4)$	d_4
(28,74)	0	0	Ir a 102
(27,75)	0	0	Ir a 102

Notar que si $x_4 = 26$ la suma máxima sería: $26 + 25 + 24 + 23 = 98 < 102$. Y que cualquier otro valor que no sea el 74 y 75 respectivamente hará que $g(X_4, d_4) \neq 0$
Solo se pondrán como posibles X_i aquellos estados desde los que se puede tomar una decisión buena (valor 0 de $g(X_k, d_k)$), puesto que solo se quieren hallar la toma de decisiones que haga que sea 0 la función objetivo.

X_3 \ d_3	(28,74)	(27,75)	$J_3(X_3)$	d_3	•
(27,47)	0	1	0	(28,74)	•
(27,48)	1	0	0	(27,75)	X
(26,48)	0	1	0	(28,74)	•
(26,49)	1	0	0	(27,75)	•
(25,49)	0	1	0	(28,74)	X

Las que pone X se descartan como estados posibles de X_3. El (27,48) porque su decisión correcta es dar valor 27 a x_4 también, incumpliendo una hipótesis. Y el (25,49) puesto que no hay dos números enteros menores que el que sumen 49 y por tanto no se podrá llegar a ese estado.

X_2 \ d_2	(27,47)	(26,48)	(26,49)	$J_2(X_2)$	d_2	•
(26,21)	0	1	2	0	(27,47)	•
(26,22)	1	0	1	0	(26,48)	X
(26,23)	2	1	0	0	(26,49)	X
(25,22)	0	1	2	0	(27,47)	•
(25,23)	1	0	1	0	(26,48)	•
(25,24)	2	1	0	0	(26,49)	•
(24,23)	0	1	2	0	(27,48)	•
(24,24)	1	0	1	0	(26,48)	X

En las que hay una X se descartan como posibles estados de X_2. Las 2 primeras porque su única decisión buena consiste en repetir el 26. Y la última X porque no hay ningún entero menor que 24 que sea 24.

X_1 \ d_1	(26,21)	(25,22)	(25,23)	(25,24)	(24,23)	$J_1(X_1)$	d_1
(24,0)	3	2	1	0	1	0	(25,24)
(23,0)	2	1	0	1	0	0	(25,23) y (24,23)
(22,0)	1	0	1	2	1	0	(25,22)
(21,0)	0	1	2	3	2	0	(26,21)

Luego fijándome únicamente en los valores de x_i que son los que interesan tenemos que los posibles valores de estos son:

-(21,26,27,28)

-(22,25,27,28)

-(23,24,27,28)

-(23,25,26,28)

-(24,25,26,27)

Que unidos a (29,30,31,32) dan los 5 posibles bombos 4.

Anexo B

ANEXO II: Programas informáticos

A continuación pondré una serie de programas informáticos. Los usados para obtener las muestras que se han usado para comparar los sorteos, es decir, los 4 tipos de sorteos, y un ejemplo de como se obtendrían las soluciones de una cierta distribución.
Cuando en un programa aparezca: ..... significará que el programa debería continuar justo debajo pero que por razones de espacio se ha tenido que mover. Y en el caso de que aparezca: .....P empezará justo debajo pero un puesto más a la izquierda por la misma razón de antes.

B.1. Sorteo FIFA

```
program sorteFIFA;
var
r1,r2,r8,r9,r10,r11,r12,r13,r14,r15,r23,r24:integer;
rr1,rr2,rr8,rr9,rr10,rr11,rr12,rr13,rr14,rr15,rr23,rr24:integer;
lista2,lista21,lista3:array[1..8] of integer;
lista1,lista11:array[1..7] of integer;
lista4:array[1..9]of integer;
listaS:array[1..4] of integer;
g:array[1..32]of integer;
g1:array[1..16]of integer;
t,t1,t2,t3,t4,t5,t6,t7,t12,peso,peso1,peso2,peso3,peso4,peso5:integer;
aux,aux1:integer;
f:text;
pos:array[1..24]of integer;
p1,p2,p3,p4,p5,p6,p7,p8,p12:integer;
p22,p32,p42,p52,p62,p72,p82,q1,p13,p14:integer;
listaa:array[1..8]of integer;
z1,z2,z3,z4,z5,z6,z7,n,n1:integer;
n2,n3,n4,n5,n6,n7,n8,c1,c2,c3,c4,c5,c6,c7,c8:integer
begin
 assign(f,'sorteoFIFA.txt');
 randomize;
 rewrite(f);
        for t1:=1 to 7 do lista1[t1]:=t1+1;
        lista11[1]:=0;
        lista11[2]:=0;
        lista11[3]:=1;
        lista11[4]:=1;
        lista11[5]:=0;
        lista11[6]:=1;
        lista11[7]:=0;
```

```
lista2[1]:=12;
lista2[2]:=17;
lista2[3]:=21;
lista2[4]:=22;
lista2[5]:=25;
lista2[6]:=26;
lista2[7]:=32;
lista2[8]:=0;
lista21[1]:=1;
lista21[2]:=20;
lista21[3]:=1;
lista21[4]:=20;
lista21[5]:=20;
lista21[6]:=20;
lista21[7]:=20;
lista21[8]:=0;
lista3[1]:=13;
lista3[2]:=23;
lista3[3]:=24;
lista3[4]:=27;
lista3[5]:=28;
lista3[6]:=29;
lista3[7]:=30;
lista3[8]:=31;
lista4[1]:=9;
lista4[2]:=10;
lista4[3]:=11;
lista4[4]:=14;
lista4[5]:=15;
lista4[6]:=16;
lista4[7]:=18;
lista4[8]:=19;
lista4[9]:=20;
listaS[1]:=1;
listaS[2]:=0;
listaS[3]:=0;
listaS[4]:=0;
for t1:=1 to 8 do
 begin
  pos[3*(t1-1) +1]:=2;
  pos[3*(t1-1) +2]:=3;
  pos[3*(t1-1) +3]:=4;
 end;
for t1:=1 to 8 do listaa[t1]:=t1;
r1:= random(9)+1;
rr1:= lista4[r1];
lista4[r1]:=lista4[9];
lista2[8]:=rr1;
g[1]:=1;
g1[1]:=1;
for t1:=1 to 7 do
 begin
  r2:=random(8-t1)+1;
  rr2:=lista1[r2];
  g[t1+1]:=rr2;
  g1[t1+1]:=lista11[r2];
  lista1[r2]:=lista1[8-t1];
  lista11[r2]:=lista11[8-t1];
 end;
aux:=2;
for t:=2 to 8 do
 begin
  if (g1[t] = 1 ) then
```

```
      begin
       listaS[aux]:=t;
       aux:=aux+1;
      end
    end;
  r8:=random(4) +1;
  rr8:=listaS[r8];
  p1:=random(3) +1;
  p12:=pos[3*(rr8-1)+p1];
  pos[3*(rr8-1)+p1]:=4;
  g[8*(p12-1) + rr8] := lista2[8];
  for t1:=rr8 to 8 do listaa[t1]:=t1+1;
  r9:=random(7)+1;
  rr9:=lista2[r9];
  c1:=lista21[r9];
  lista2[r9]:=lista2[7];
  lista21[r9]:=lista21[7];
  aux1:=0;
  if (c1 = 20)then
   begin
    z1:=listaa[1];
    for t1:=1 to 6 do listaa[t1]:=listaa[t1+1];
    p2:=random(3)+1;
    p22:=pos[3*(z1-1)+p2];
    pos[3*(z1-1)+p2]:=4;
    g[8*(p22-1) +z1]:=rr9;
   end;
  if (c1 = 1)then
   begin
    t2:=1;
    aux1:=aux1+1;
    while t2<= 7 do
     begin
      n:=listaa[t2];
      if (g1[n] = 0)then
       begin
        p2:=random(3)+1;
        p22:=pos[3*(n-1)+p2];
        pos[3*(n-1)+p2]:=4;
        g[8*(p22-1)+n]:=rr9;
        for t3:=t2 to 7 do listaa[t3]:=listaa[t3+1];
        t2:=8;
       end;
      t2:=t2+1;
     end;
   end;
  r10:=random(6)+1;
  rr10:=lista2[r10];
  c2:=lista21[r10];
  lista2[r10]:=lista2[6];
  lista21[r10]:=lista21[6];
  if (c2 = 20) then
   begin
    z2:=listaa[1];
    for t1:=1 to 5 do listaa[t1]:=listaa[t1+1];
    p3:=random(3)+1;
    p32:=pos[3*(z2-1)+p3];
    pos[3*(z2-1)+p3]:=4;
    g[8*(p32-1) +z2]:=rr10;
   end;
  if (c2 = 1)then
   begin
    t4:=1;
```

```
  aux1:=aux1+1;
  while t4<= 6 do
   begin
    n1:=listaa[t4];
    if (g1[n1] = 0)then
     begin
      p3:=random(3)+1;
      p32:=pos[3*(n1-1)+p3];
      pos[3*(n1-1)+p3]:=4;
      g[8*(p32-1)+n1]:=rr10;
      for t5:=t4 to 6 do listaa[t5]:=listaa[t5+1];
      t4:=7;
     end;
    t4:=t4+1;
   end;
 end;
n2:=0;
peso:=listaa[1];
peso1:=listaa[2];
peso2:=listaa[3];
peso3:=listaa[4];
peso4:=listaa[5];
n2:= g1[peso]+g1[peso1]+g1[peso2]+g1[peso3]+g1[peso4];
r11:=random(5)+1;
rr11:=lista2[r11];
c3:=lista21[r11];
lista2[r11]:=lista2[5];
lista21[r11]:=lista21[5];
if (c3 = 20) then
 begin
  if(5-n2 > 2-aux1) then
   begin
    z3:=listaa[1];
    for t1:=1 to 4 do listaa[t1]:=listaa[t1+1];
    p4:=random(3)+1;
    p42:=pos[3*(z3-1)+p4];
    pos[3*(z3-1)+p4]:=4;
    g[8*(p42-1) +z3]:=rr11;
   end;
  if (5-n2 = 2-aux1) then
   begin
    t6:=1;
    while t6<=5 do
     begin
      n3:= listaa[t6];
      if (g1[n3] = 1) then
       begin
        p4:=random(3)+1;
        p42:=pos[3*(n3-1)+p4];
        pos[3*(n3-1)+p4]:=4;
        g[8*(p42-1)+n3]:=rr11;
        for t7:=t6 to 5 do listaa[t7]:=listaa[t7+1];
        t6:=6;
       end;
      t6:=t6+1;
     end;
   end;
 end;
if (c3 = 1) then
 begin
  aux1:=aux1+1;
  t6:=1;
  while t6<=5 do
```

```
      begin
       n3:=listaa[t6];
       if (g1[n3] = 0) then
        begin
         p4:=random(3)+1;
         p42:=pos[3*(n3-1)+p4];
         pos[3*(n3-1)+p4]:=4;
         g[8*(p42-1)+n3]:=rr11;
         for t7:=t6 to 5 do listaa[t7]:=listaa[t7+1];
         t6:=6;
        end;
       t6:=t6+1;
      end;
    end;
  peso:=listaa[1];
  peso1:=listaa[2];
  peso2:=listaa[3];
  peso3:=listaa[4];
  n2:= g1[peso]+g1[peso1]+g1[peso2]+g1[peso3];
  r12:=random(4)+1;
  rr12:=lista2[r12];
  c4:=lista21[r12];
  lista2[r12]:=lista2[4];
  lista21[r12]:=lista21[4];
  if (c4 = 20) then
   begin
    if(4-n2 > 2-aux1) then
     begin
      z4:=listaa[1];
      for t1:=1 to 3 do listaa[t1]:=listaa[t1+1];
      p5:=random(3)+1;
      p52:=pos[3*(z4-1)+p5];
      pos[3*(z4-1)+p5]:=4;
      g[8*(p52-1) +z4]:=rr12;
     end;
    if (4-n2 = 2-aux1)then
     begin
      t6:=1;
      while t6<=4 do
       begin
        n3:= listaa[t6];
        if (g1[n3] = 1) then
         begin
          p5:=random(3)+1;
          p52:=pos[3*(n3-1)+p5];
          pos[3*(n3-1)+p5]:=4;
          g[8*(p52-1)+n3]:=rr12;
          for t7:=t6 to 4 do listaa[t7]:=listaa[t7+1];
          t6:=5;
         end;
        t6:=t6+1;
       end;
     end;
   end;
  if (c4 = 1) then
   begin
    t6:=1;
    aux1:=aux1+1;
    while t6<=4 do
     begin
      n3:=listaa[t6];
      if (g1[n3] = 0) then
       begin
```

```
        p5:=random(3)+1;
        p52:=pos[3*(n3-1)+p5];
        pos[3*(n3-1)+p5]:=4;
        g[8*(p52-1)+n3]:=rr12;
        for t7:=t6 to 4 do listaa[t7]:=listaa[t7+1];
        t6:=5;
       end;
      t6:=t6+1;
     end;
 end;
peso:=listaa[1];
peso1:=listaa[2];
peso2:=listaa[3];
n2:= g1[peso]+g1[peso1]+g1[peso2];
r13:=random(3)+1;
rr13:=lista2[r13];
c5:=lista21[r13];
lista2[r13]:=lista2[3];
lista21[r13]:=lista21[3];
if (c5 = 20) then
 begin
  if(3-n2 > 2-aux1) then
   begin
    z5:=listaa[1];
    for t1:= 1 to 2 do listaa[t1]:=listaa[t1+1];
    p6:=random(3)+1;
    p62:=pos[3*(z5-1)+p6];
    pos[3*(z5-1)+p6]:=4;
    g[8*(p62-1) +z5]:=rr13;
   end;
  if (3-n2 = 2-aux1)then
   begin
    t6:=1;
    while t6<=3 do
     begin
      n3:= listaa[t6];
      if (g1[n3] = 1) then
       begin
        p6:=random(3)+1;
        p62:=pos[3*(n3-1)+p6];
        pos[3*(n3-1)+p6]:=4;
        g[8*(p62-1)+n3]:=rr13;
        for t7:=t6 to 3 do listaa[t7]:=listaa[t7+1];
        t6:=4;
       end;
      t6:=t6+1;
     end;
   end;
 end;
if (c5 = 1) then
 begin
  t6:=1;
  aux1:=aux1+1;
  while t6<=3 do
   begin
    n3:=listaa[t6];
    if (g1[n3] = 0) then
     begin
      p6:=random(3)+1;
      p62:=pos[3*(n3-1)+p6];
      pos[3*(n3-1)+p6]:=4;
      g[8*(p62-1)+n3]:=rr13;
      for t7:=t6 to 3 do listaa[t7]:=listaa[t7+1];
```

```
        t6:=4;
       end;
      t6:=t6+1;
    end;
 end;
peso:=listaa[1];
peso1:=listaa[2];
n2:= g1[peso]+g1[peso1];
r14:=random(2)+1;
rr14:=lista2[r14];
c6:=lista21[r14];
lista2[r14]:=lista2[2];
lista21[r14]:=lista21[2];
if (c6 = 20) then
 begin
  if(2-n2 > 2-aux1) then
   begin
    z6:=listaa[1];
    listaa[1]:=listaa[2];
    p7:=random(3)+1;
    p72:=pos[3*(z6-1)+p7];
    pos[3*(z6-1)+p7]:=4;
    g[8*(p72-1) +z6]:=rr14;
   end;
  if (2-n2 = 2-aux1)then
   begin
    t6:=1;
    while t6<=2 do
     begin
      n3:= listaa[t6];
      if (g1[n3] = 1) then
       begin
        p7:=random(3)+1;
        p72:=pos[3*(n3-1)+p7];
        pos[3*(n3-1)+p7]:=4;
        g[8*(p72-1)+n3]:=rr14;
        for t7:=t6 to 2 do listaa[t7]:=listaa[t7+1];
        t6:=3;
       end;
      t6:=t6+1;
     end;
   end;
 end;
if (c6 = 1) then
 begin
  t6:=1;
  aux1:=aux1+1;
  while t6<=2 do
   begin
    n3:=listaa[t6];
    if (g1[n3] = 0) then
     begin
      p7:=random(3)+1;
      p72:=pos[3*(n3-1)+p7];
      pos[3*(n3-1)+p7]:=4;
      g[8*(p72-1)+n3]:=rr14;
      for t7:=t6 to 2 do listaa[t7]:=listaa[t7+1];
      t6:=3;
     end;
    t6:=t6+1;
   end;
 end;
rr15:=lista2[1];
```

```
        n4:=listaa[1];
        p8:=random(3)+1;
        p82:=pos[3*(n4-1)+p8];
        pos[3*(n4-1)+p8]:=4;
        g[8*(p82-1)+n4]:=rr15;
        for t4:=1 to 8 do
         begin
          r23:=random(9-t4)+1;
          rr23:=lista3[r23];
          lista3[r23]:=lista3[9-t4];
          q1:=random(2)+1;
          p13:=pos[3*(t4-1)+q1];
          pos[3*(t4-1)+q1]:=pos[2+3*(t4-1)];
          g[8*(p13-1)+t4]:=rr23;
         end;
        for t5:=1 to 8 do
         begin
          r24:= random(9-t5) +1;
          rr24:= lista4[r24];
          lista4[r24]:=lista4[9-t5];
          p14:=pos[1+3*(t5-1)];
          g[8*(p14-1)+t5]:=rr24;
         end;
        writeLn(f,'G1:',g[1],',',g[9],',',g[17],',',g[25],'-');
        writeLn(f,'G2:',g[2],',',g[10],',',g[18],',',g[26],'-');
        writeLn(f,'G3:',g[3],',',g[11],',',g[19],',',g[27],'-');
        writeLn(f,'G4:',g[4],',',g[12],',',g[20],',',g[28],'-');
        writeLn(f,'G5:',g[5],',',g[13],',',g[21],',',g[29],'-');
        writeLn(f,'G6:',g[6],',',g[14],',',g[22],',',g[30],'-');
        writeLn(f,'G7:',g[7],',',g[15],',',g[23],',',g[31],'-');
        writeLn(f,'G8:',g[8],',',g[16],',',g[24],',',g[32],'-');
 close(f);
end.
```

B.2. Sorteo de Guyon

Sorteo Guyon. Los documentos disguy1.txt y disguy2.txt es donde están almacenadas las soluciones geográficas. Asimismo la solución al sorteo se almacenará en el documento sorGuyon.txt

```
program sorn;
var
f:text;


Procedure sorteo1;
var
i,j,k,l,m,n,o,p,r,t,c:integer;
a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11,a12:integer;
r1,r2,r3,r4,r5,r6,r7,r8,r9:integer;
rr1,rr2,rr3,rr4,rr5,rr6,rr7,rr8,rr9:integer;
t1,t2,t3,t4,t5,t6,t7,t8,t9,z:integer;
v3,v4:real;
f1: text;
g,g1:array[1..16] of integer;
h:array[1..96] of integer;
lis1,lis2,lis3,lis4:array[1..5] of integer;
lista1, lista2,lista3,lista4,lista5,lista6:array[1..3]of integer;
begin
     assign(f1,'disguy1.txt');
     reset(f1);
     c:=0;
     while not Eoln(f1) do
     begin
      Inc(c);
      Read(f1,h[c]);
     end;
     r:=1 + random(6);
     for t:=1 to 4 do g1[4*(t-1) +1]:=h[t+(r-1)*16];
     for t:=1 to 4 do g1[4*(t-1) +2]:=h[t+4+(r-1)*16];
     for t:=1 to 4 do g1[4*(t-1) +3]:=h[t+8+(r-1)*16];
     for t:=1 to 4 do g1[4*(t-1) +4]:=h[t+12+(r-1)*16];
     g[1]:=1;
     g[2]:=2;
     g[3]:=3;
     g[4]:=4;
     lis1[1]:=1;
     lis1[2]:=2;
     lis1[3]:=3;
     lis1[4]:=4;
     lis1[5]:=67;
     for i:= 1 to 4 do
      begin
       if(g1[4+i]=5)then
        begin
         g[4+i]:=13;
         lis2:=lis1;
         for z:=i to 4 do lis2[z]:=lis2[z+1];
         t1:=lis2[1];
         t2:=lis2[2];
         t3:=lis2[3];
         lista1[1]:=14;
         lista1[2]:=15;
         lista1[3]:=16;
         r2:=random(3)+1;
         rr2:=lista1[r2];
```

```
g[4+t1]:=rr2;
lista2:=lista1;
lista2[r2]:=lista1[3];
r3:=random(2)+1;
rr3:=lista2[r3];
g[4+t2]:=rr3;
lista2[r3]:=lista2[2];
g[4+t3]:=lista2[1];
for j:= 1 to 4 do
 begin
  if(g1[8+j]=20)then
   begin
    g[8+j]:=17;
    lis3:=lis1;
    for z:=j to 4 do lis3[z]:=lis3[z+1];
    t4:=lis3[1];
    t5:=lis3[2];
    t6:=lis3[3];
    lista3[1]:=18;
    lista3[2]:=19;
    lista3[3]:=20;
    r4:=random(3)+1;
    rr4:=lista3[r4];
    g[8+t4]:=rr4;
    lista4:=lista3;
    lista4[r4]:=lista3[3];
    r5:=random(2)+1;
    rr5:=lista4[r5];
    g[8+t5]:=rr5;
    lista4[r5]:=lista4[2];
    g[8+t6]:=lista4[1];
    for k:= 1 to 4 do
     begin
      if(g1[12+k]=20)then
       begin
        g[12+k]:=32;
        lis4:=lis1;
        for z:=k to 4 do lis4[z]:=lis4[z+1];
        t7:=lis4[1];
        t8:=lis4[2];
        t9:=lis4[3];
        lista5[1]:=29;
        lista5[2]:=30;
        lista5[3]:=31;
        r6:=random(3)+1;
        rr6:=lista5[r6];
        g[12+t7]:=rr6;
        lista6:=lista5;
        lista6[r6]:=lista5[3];
        r7:=random(2)+1;
        rr7:=lista6[r7];
        g[12+t8]:=rr7;
        lista6[r7]:=lista6[2];
        g[12+t9]:=lista6[1];
        writeLn(f,'G1:',g[1],',',g[5],',',g[9],',',g[13],'--');
        writeLn(f,'G2:',g[2],',',g[6],',',g[10],',',g[14],'--');
        writeLn(f,'G3:',g[3],',',g[7],',',g[11],',',g[15],'--');
        writeLn(f,'G4:',g[4],',',g[8],',',g[12],',',g[16]);
       end;
     end;
   end;
 end;
end;
```

```
      end;
close(f1);
end;

Procedure sorteo2;
var
i,j,k,l,m,n,o,p,c,r,t:integer;
a1,a2,a3,a4,a5,a6,a7,a8,a9,a10,a11,a12:integer;
r2,r3,r4,r5,r6,r7,r8,r9:integer;
rr1,rr2,rr3,rr4,rr5,rr6,rr7,rr8,rr9:integer;
t1,t2,t3,t4,t5,t6,t7,t8,t9,z:integer;
v3,v4:real;
f1:text;
h:array[1..384] of integer;
g,g1:array[1..16] of integer;
lis1,lis2,lis3,lis4,lis5,lis6,lis7:array[1..5] of integer;
lista1, lista2,lista3,lista4,lista5,lista6:array[1..3]of integer;
r1:integer;

begin
     assign(f1,'disguy2.txt');
     reset(f1);
     c:=0;
     while not Eoln(f1) do
     begin
      Inc(c);
      Read(f1,h[c]);
     end;
     r:=1 + random(24);
     for t:=1 to 4 do g1[4*(t-1) +1]:=h[t+(r-1)*16];
     for t:=1 to 4 do g1[4*(t-1) +2]:=h[t+4+(r-1)*16];
     for t:=1 to 4 do g1[4*(t-1) +3]:=h[t+8+(r-1)*16];
     for t:=1 to 4 do g1[4*(t-1) +4]:=h[t+12+(r-1)*16];
     g[1]:=5;
     g[2]:=6;
     g[3]:=7;
     g[4]:=8;
     lis1[1]:=1;
     lis1[2]:=2;
     lis1[3]:=3;
     lis1[4]:=4;
     lis1[5]:=67;
     for i:=1 to 4 do
      begin
       if(g1[4+i]=1)then
        begin
         g[4+i]:=12;
         lis2:=lis1;
         for z:=i to 4 do lis2[z]:=lis2[z+1];
         t1:=lis2[1];
         t2:=lis2[2];
         t3:=lis2[3];
         lista1[1]:=9;
         lista1[2]:=10;
         lista1[3]:=11;
         r2:=random(3)+1;
         rr2:=lista1[r2];
         g[4+t1]:=rr2;
         lista2:=lista1;
         lista2[r2]:=lista1[3];
         r3:=random(2)+1;
         rr3:=lista2[r3];
         g[4+t2]:=rr3;
```

```
          lista2[r3]:=lista2[2];
          g[4+t3]:=lista2[1];
          for j:= 1 to 4 do
           begin
            if(g1[8+j]=1)then
             begin
              g[8+j]:=21;
              lis3:=lis1;
              for z:=j to 4 do lis3[z]:=lis3[z+1];
              for k:=1 to 3 do
               begin
                r8:=lis3[k];
                if(g1[8+r8]=20)then
                 begin
                  g[8+r8]:=22;
                  lis4:=lis3;
                  for z:=k to 4 do lis4[z]:=lis4[z+1];
                  lista3[1]:=23;
                  lista3[2]:=24;
                  t4:=lis4[1];
                  t5:=lis4[2];
                  r4:=random(2)+1;
                  rr4:=lista3[r4];
                  g[8+t4]:=rr4;
                  lista3[r4]:=lista3[2];
                  g[8+t5]:=lista3[1];
                  for l:=1 to 4 do
                   begin
                    if(g1[12+l]=50)then
                     begin
                      g[12+l]:=28;
                      lis5:=lis1;
                      for z:=1 to 4 do lis5[z]:=lis5[z+1];
                      for m:=1 to 3 do
                       begin
                        r9:=lis5[m];
                        if(g1[12+r9]=5)then
                         begin
                          g[12+r9]:=27;
                          lis6:=lis5;
                          for z:=m to 4 do lis6[z]:=lis6[z+1];
                          lista4[1]:=25;
                          lista4[2]:=26;
                          t6:=lis6[1];
                          t7:=lis6[2];
                          r5:=random(2)+1;
                          rr5:=lista4[r5];
                          g[12+t6]:= rr5;
                          lista4[r5]:=lista4[2];
                          g[12+t7]:=lista4[1];
                          .....
writeLn(f,'G5:',g[1],',',g[5],',',g[9],',',g[13],'—');
writeLn(f,'G6:',g[2],',',g[6],',',g[10],',',g[14],'—');
writeLn(f,'G7:',g[3],',',g[7],',',g[11],',',g[15],'—');
writeLn(f,'G8:',g[4],',',g[8],',',g[12],',',g[16]);
.....P
                         end;
                       end;
                     end;
                   end;
                 end;
               end;
             end;
```

```
            end;
          end;
        end;
close(f1);
end;

begin
assign(f,'sorGUYON.txt');
rewrite(f);
randomize;

sorteo1;
sorteo2;

close(f);
end.
```

B.3. Agrupaciones perfectas

Para hallar todas las soluciones que genera una distribución, divido este proceso en 2 programas. El primero encuentra las agrupaciones incompletas (sin el 4º equipo) que cumplan que pueden ser soluciones. Y el siguiente programa completará los grupos de forma que sean soluciones. En este caso buscaremos las soluciones que son generadas por la distribución básica B4=B4A.

```
program agrupaciones_2428basico;

uses crt;
var
ii,jj,kk,ll,mm,nn,oo,ii1,jj1,kk1,ll1,mm1,nn1,oo1, aux, aux1:integer;
I,J,K,L,M,N,O,P,I1,J1,K1,L1,M1,N1,O1,P1: integer;
pes1,pes2,pes3,pes4,pes5,pes6,pes7,pes8,t:integer;
var
lis1, vec1,liis1: array[1..4] of integer ;
lis2,lis3,lis4,lis5,lis6,lis7,lis8 :array[1..8] of integer ;
vec2,vec3,vec4,vec5,vec6,vec7,vec8,liis:array[1..8] of integer ;

g: array[1..24] of integer;
g1: array[1..24] of integer;


begin
    clrscr;
    g[1]:=1;
    g[2]:=2;
    g[3]:=3;
    g[4]:=4;
    g[5]:=5;
    g[6]:=6;
    g[7]:=7;
    g[8]:=8;
    g1[1]:=1;
    g1[2]:=0;
    g1[3]:=0;
    g1[4]:=1;
    g1[5]:=1;
    g1[6]:=0;
    g1[7]:=1;
    g1[8]:=0;
    liis1[1]:=2;
    liis1[2]:=3;
    liis1[3]:=6;
    liis1[4]:=8;
    vec1:=liis1;
    lis1:=liis1;
    for t:=1 to 8 do
     liis[t]:=t;
    lis2:=liis;
    lis3:=liis;
    lis4:=liis;
    lis5:=liis;
    lis6:=liis;
    lis7:=liis;
    lis8:=liis;
    vec2:=liis;
    vec3:=liis;
    vec4:=liis;
    vec5:=liis;
    vec6:=liis;
    vec7:=liis;
```

```
    vec8:=liis;

    for ii:=1 to 4 do
     begin
      I:=lis1[ii];
      g[8+I]:=12;
      g1[8+I]:=1;
      pes1:=g[I]+12;
      lis2:=liis;
      lis2[I]:=8;
      for jj:=1 to 5 do
       begin
        J:=lis2[jj];
        pes2:=g[J]+16;
        if ((pes1-pes2)*(J-6)*(J-8)<>0) then
         begin
          g[8+J]:=16;
          g1[8+J]:=0;
          lis3:=lis2;
          lis3[jj]:=lis2[7];
          for kk:=1 to 6 do
           begin
            K:=lis3[kk];
            pes3:=g[K]+15;
            if ((pes1-pes3)*(pes2-pes3)*(K-7)*(K-8)<>0) then
             begin
              g[8+K]:=15;
              g1[8+K]:=0;
              lis4:=lis3;
              lis4[kk]:=lis3[6];
              for ll:=1 to 5 do
               begin
                L:=lis4[ll];
                pes4:=g[L]+9;
                if ((pes1-pes4)*(pes2-pes4)*(pes3-pes4)<>0) then
                 begin
                  g[L+8]:=9;
                  g1[L+8]:=0;
                  lis5:=lis4;
                  lis5[ll]:=lis4[5];
                  for mm:=1 to 4 do
                   begin
                    M:=lis5[mm];
                    pes5:=g[M]+10;
                    .....
if ((pes1-pes5)*(pes2-pes5)*(pes3-pes5)*(pes4-pes5)<>0)then
 begin
  g[8+M]:=10;
  g1[8+M]:=0;
  lis6:=lis5;
  lis6[mm]:=lis5[4];
  for nn:=1 to 3 do
   begin
    N:=lis6[nn];
    pes6:=g[N]+11;
        .....
if ((pes1-pes6)*(pes2-pes6)*(pes3-pes6)*(pes4-pes6)*(pes5-pes6)<>0)then
 begin
  g[8+N]:=11;
  g1[8+N]:=0;
  lis7:=lis6;
  lis7[nn]:=lis6[3];
  for oo:=1 to 2 do
```

```
begin
 O:=lis7[oo];
 pes7:= 13+g[O];
 aux:=(pes1-pes7)*(pes2-pes7)*(pes3-pes7)*(pes4-pes7);
 if(aux*(pes5-pes7)*(pes6-pes7)<>0) then
  begin
   g[O+8]:=13;
   g1[O+8]:=5;
   lis8:=lis7;
   lis8[oo]:=lis7[2];
   P:=lis8[1];
   pes8:=g[P]+14;
   aux1:=(pes1-pes8)*(pes2-pes8)*(pes3-pes8)*(pes4-pes8);
   if(aux1*(pes5-pes8)*(pes6-pes8)*(pes7-pes8)<>0) then
    begin
     g[P+8]:=14;
     g1[P+8]:=0;
     for ii1:=1 to 3 do
      begin
       vec1:=liis1;
       vec1[ii]:=8;
       I1:=vec1[ii1];
       if(g[I1]+g[I1+8]+21 =38) then
        begin
         g[I1+16]:=21;
         g1[I1+16]:=1;
         vec2:=liis;
         vec2[I1]:=8;
         for jj1:=1 to 5 do
          begin
           J1:=vec2[jj1];
           if(g[J1]+g[J1+8]+28=38) then
            begin
             g[J1+16]:=28;
             g1[J1+16]:=50;
             vec3:=vec2;
             vec3[jj1]:=vec2[7];
             for kk1:=1 to 6 do
              begin
               K1:=vec3[kk1];
               if(g[K1]+g[8+K1]+22 = 38) then
                begin
                 g[K1+16]:=22;
                 g1[K1+16]:=20;
                 vec4:=vec3;
                 vec4[kk1]:=vec3[6];
                 for ll1:=1 to 5 do
                  begin
                   L1:=vec4[ll1];
                   if(g[L1]+g[L1+8]+17 = 38) then
                    begin
                     g[L1+16]:=17;
                     g1[L1+16]:=20;
                     vec5:=vec4;
                     vec5[ll1]:=vec4[5];
                     for mm1:=1 to 4 do
                      begin
                       M1:=vec5[mm1];
                       if(g[M1]+g[M1+8]+18 = 38) then
                        begin
                         g[M1+16]:=18;
                         g1[M1+16]:=0;
                         vec6:=vec5;
```

```
                                vec6[mm1]:=vec5[4];
                                for nn1:=1 to 3 do
                                 begin
                                  N1:= vec6[nn1];
                                  if(g[N1]+g[N1+8]+19 = 38) then
                                   begin
                                    g[N1+16]:=19;
                                    g1[N1+16]:=0;
                                    vec7:=vec6;
                                    vec7[nn1]:=vec6[3];
                                    for oo1:=1 to 2 do
                                     begin
                                      O1:=vec7[oo1];
                                      if(g[O1]+g[O1+8]+20 = 38) then
                                       begin
                                        g[O1+16]:=20;
                                        g1[O1+16]:=0;
                                        vec8:=vec7;
                                        vec8[oo1]:=vec7[2];
                                        P1:=vec8[1];
                                        g[P1+16]:=23;
                                        g1[P1+16]:=5;
                                        .....
if((g1[2]+g1[10]+g1[18])*(g1[3]+g1[11]+g1[19])*(g1[6]+g1[14]+g1[22])*
(g1[8]+g1[16]+g1[24])*(g1[O+8]+g1[O+16]-10)<>0) then
 begin
  write('G1:',g[1],',',g[9],',',g[17],'---');
  write('G2:',g[2],',',g[10],',',g[18],'---');
  write('G3:',g[3],',',g[11],',',g[19],'---');
  write('G4:',g[4],',',g[12],',',g[20]);
  writeln('');
  write('G5:',g[5],',',g[13],',',g[21],'---');
  write('G6:',g[6],',',g[14],',',g[22],'---');
  write('G7:',g[7],',',g[15],',',g[23],'---');
  write('G8:',g[8],',',g[16],',',g[24]);
  writeln('');
  .....
                                                                    writeln('');
                                                                   end;
                                                                  end;
                                                                 end;
                                                                end;
                                                               end;
                                                              end;
                                                             end;
                                                            end;
                                                           end;
                                                          end;
                                                         end;
                                                        end;
                                                       end;
                                                      end;
                                                     end;
                                                    end;
                                                   end;
                                                  end;
                                                 end;
                                                end;
                                               end;
                                              end;
                                             end;
                                            end;
                                           end;
                                          end;
                                         end;
                                        end;
                                       end;
                                      end;
                                     end;
                                    end;
                                   end;
                                  end;
                                 end;
                                end;
                               end;
                              end;
                             end;
                            end;
                           end;
                          end;
                         end;
                        end;
                       end;
                      end;
                     end;
                    end;
                   end;
                  end;
                 end;
                end;
               end;
              end;
             end;
            end;
           end;
          end;
         end;
        end;
       end;
      end;
     end;
    end;
   end;
  end;
 end;
end;
```

```
                end;

            end;
        end;
    end;

    readkey;
end.
```

Programa que completa el anterior para acabar dando todas las soluciones que genera la distribución básica B4=B4A.

```
program gruposperfectoslos4;
uses crt;
var
ii, I, jj, J, kk, K, ll, L, mm, M, nn, N, oo, O, pp, P, t, t1, t2:integer;
var
f:text;
var
lis1,lis2,lis3,lis4,lis5,lis6,lis7,lis8,lis: array[1..8] of integer;

g,g1: array[1..32] of integer;
begin
   clrscr;
   assign(f,'listabas2428.txt');
   rewrite(f);
   g[1]:=1;
   g[2]:=2;
   g[3]:=3;
   g[4]:=4;
   g[5]:=5;
   g[6]:=6;
   g[7]:=7;
   g[8]:=8;
   g[9]:=9;
   g[10]:=15;
   g[11]:=12;
   g[12]:=16;
   g[13]:=14;
   g[14]:=10;
   g[15]:=11;
   g[16]:=13;
   g[17]:=28;
   g[18]:=21;
   g[19]:=23;
   g[20]:=18;
   g[21]:=19;
   g[22]:=22;
   g[23]:=20;
   g[24]:=17;
   g1[1]:=1;
   g1[2]:=0;
   g1[3]:=0;
   g1[4]:=1;
   g1[5]:=1;
   g1[6]:=0;
   g1[7]:=1;
   g1[8]:=0;
   g1[9]:=0;
   g1[10]:=0;
   g1[11]:=1;
   g1[12]:=0;
   g1[13]:=0;
   g1[14]:=0;
   g1[15]:=0;
   g1[16]:=5;
   g1[17]:=50;
   g1[18]:=1;
   g1[19]:=5;
   g1[20]:=0;
   g1[21]:=0;
   g1[22]:=20;
```

```
g1[23]:=0;
g1[24]:=20;
lis[1]:=1;
lis[2]:=2;
lis[3]:=3;
lis[4]:=4;
lis[5]:=5;
lis[6]:=6;
lis[7]:=7;
lis[8]:=8;
for ii:=1 to 8 do
 begin
  lis1:=lis;
  I:=lis1[ii];
  if(24>g[16+I]) then
   begin
    g[24+I]:=24;
    g1[24+I]:=5;
    lis2:=lis1;
    lis2[ii]:=lis1[8];
    for jj:=1 to 7 do
     begin
      J:=lis2[jj];
      if(25>g[16+J]) then
       begin
        g[24+J]:=25;
        g1[24+J]:=20;
        lis3:=lis2;
        lis3[jj]:=lis2[7];
        for kk:=1 to 6 do
         begin
          K:=lis3[kk];
          g[24+K]:=26;
          g1[24+K]:=20;
          lis4:=lis3;
          lis4[kk]:=lis3[6];
          for ll:=1 to 5 do
           begin
            L:=lis4[ll];
            g[24+L]:=27;
            g1[24+L]:=5;
            lis5:=lis4;
            lis5[ll]:=lis4[5];
            for mm:=1 to 4 do
             begin
              M:=lis5[mm];
              g[24+M]:=29;
              g1[24+M]:=50;
              lis6:=lis5;
              lis6[mm]:=lis5[4];
              for nn:=1 to 3 do
               begin
                N:=lis6[nn];
                g[24+N]:=30;
                g1[24+N]:=50;
                lis7:=lis6;
                lis7[nn]:=lis6[3];
                for oo:=1 to 2 do
                 begin
                  O:=lis7[oo];
                  g[24+O]:=31;
                  g1[24+O]:=50;
                  lis8:=lis7;
```

```
                        lis8[oo]:=lis7[2];
                        P:=lis8[1];
                        g[24+P]:=32;
                        g1[24+P]:=20;
                        t:=(g1[16+P]-20)*(g1[8+P]-20);
                        t1:=(g[L+16]-28)*(g1[16+L]-5);
                        t2:=(g1[16+J]-20)*(g1[8+J]-20)
                        .....
if(t*(g1[16+O]-50)*(g1[16+N]-50)*(g1[16+M]-50)<>0)then
 begin
  if(t1*(g1[8+L]-5)*(g[16+K]-28)*(g1[8+K]-20)*(g1[16+K]-20)<>0)then
   begin
    if(t2*(g1[8+I]-5)*(g1[16+I]-5)<>0)then
     begin
      writeLn(f,'G1:',g[1],',',g[9],',',g[17],',',g[25]);
      writeLn(f,'G2:',g[2],',',g[10],',',g[18],',',g[26]);
      writeLn(f,'G3:',g[3],',',g[11],',',g[19],',',g[27]);
      writeLn(f,'G4:',g[4],',',g[12],',',g[20],',',g[28]);
      writeLn(f,'G5:',g[5],',',g[13],',',g[21],',',g[29]);
      writeLn(f,'G6:',g[6],',',g[14],',',g[22],',',g[30]);
      writeLn(f,'G7:',g[7],',',g[15],',',g[23],',',g[31]);
      writeLn(f,'G8:',g[8],',',g[16],',',g[24],',',g[32]);
      writeLn(f,'');
      .....P
                            end;
                          end;
                        end;
                      end;
                    end;
                  end;
                end;
              end;
            end;
          end;
        end;
      end;
close(f);
readkey;
end.
```

B.4. Sorteo Perfecto

Sorteo Perfecto. Los documentos usados en este programa son soluciones intermedias (3 equipos en cada grupo) que luego se completarán con el B4 correspondiente formando agrupaciones perfectas. Este programa se encarga de elegir 10 soluciones (agrupaciones perfectas) al azar.

```
program sorteoperfecto;
var
num,num1:real;
var3,var4:real;
datos,datos1,datos2,lista:text;
aa,bb:array[1..8] of real;
rango4:real;
con:integer;
v4A,v4B,v4C,v4D,v4E:real;
m3A,m3B,m3C,m3D,m3E,m4A,m4B,m4C,m4D,m4E:array[1..8] of real;
gruposA,gruposB,gruposC,gruposD,gruposE:array[1..8] of real;

procedure bA;
var
ii, I, jj, J, kk, K, ll, L, mm, M, nn, N, oo, O, pp, P:integer;
t,t1,c,c1,c2:integer;
var
lis1,lis2,lis3,lis4,lis5,lis6,lis7,lis8,lis: array[1..8] of integer;
f,f1,f2:text;
h: array[1..48] of integer;
h1,h2: array[1..192] of integer;

g,g1: array[1..32] of integer;
begin
 assign(f,'geo2125b.txt');
 assign(f1,'geo2125c1.txt');
 assign(f2,'geo2125c2.txt');
 reset(f);
 reset(f1);
 reset(f2);
 c:=0;
 c1:=0;
 c2:=0;
 while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
 while not Eoln(f1) do
  begin
   Inc(c1);
   Read(f1,h1[c1]);
  end;
 while not Eoln(f2) do
  begin
   Inc(c2);
   Read(f2,h2[c2]);
  end;

 for t:= 1 to 9 do
  begin
   case t of
    1 : begin
         for t1:=1 to 24 do g[t1]:=h[t1];
         for t1:=1 to 24 do g1[t1]:=h[t1+24];
        end;
```

```
  2 : begin
       for t1:=1 to 24 do g[t1]:= h1[t1];
       for t1:=1 to 24 do g1[t1]:= h1[t1+24];
      end;
  3 : begin
       for t1:= 1 to 24 do g[t1]:= h1[t1+48];
       for t1:= 1 to 24 do g1[t1]:=h1[t1+72];
      end;
  4 : begin
       for t1:= 1 to 24 do g[t1]:= h1[t1+96];
       for t1:= 1 to 24 do g1[t1]:=h1[t1+120];
      end;
  5 : begin
       for t1:= 1 to 24 do g[t1]:= h1[t1+144];
       for t1:= 1 to 24 do g1[t1]:=h1[t1+168];
      end;
  6 : begin
       for t1:= 1 to 24 do g[t1]:= h2[t1];
       for t1:= 1 to 24 do g1[t1]:=h2[t1+24];
      end;
  7 : begin
       for t1:= 1 to 24 do g[t1]:= h2[t1+48];
       for t1:= 1 to 24 do g1[t1]:=h2[t1+72];
      end;
  8 : begin
       for t1:= 1 to 24 do g[t1]:= h2[t1+96];
       for t1:= 1 to 24 do g1[t1]:=h2[t1+120];
      end;
  9 : begin
       for t1:= 1 to 24 do g[t1]:= h2[t1+144];
       for t1:= 1 to 24 do g1[t1]:=h2[t1+168];
      end;
end;
lis[1]:=1;
lis[2]:=2;
lis[3]:=3;
lis[4]:=4;
lis[5]:=5;
lis[6]:=6;
lis[7]:=7;
lis[8]:=8;
for ii:=1 to 8 do
 begin
  lis1:=lis;
  I:=lis1[ii];
  if(21>g[16+I]) then
   begin
    g[24+I]:=21;
    g1[24+I]:=1;
    lis2:=lis1;
    lis2[ii]:=lis1[8];
    for jj:=1 to 7 do
     begin
      J:=lis2[jj];
      if(26>g[16+J]) then
       begin
        g[24+J]:=26;
        g1[24+J]:=20;
        lis3:=lis2;
        lis3[jj]:=lis2[7];
        for kk:=1 to 6 do
         begin
          K:=lis3[kk];
```

```
                g[24+K]:=27;
                g1[24+K]:=5;
                lis4:=lis3;
                lis4[kk]:=lis3[6];
                for ll:=1 to 5 do
                 begin
                  L:=lis4[ll];
                  g[24+L]:=28;
                  g1[24+L]:=50;
                  lis5:=lis4;
                  lis5[ll]:=lis4[5];
                  for mm:=1 to 4 do
                   begin
                    M:=lis5[mm];
                    g[24+M]:=29;
                    g1[24+M]:=50;
                    lis6:=lis5;
                    lis6[mm]:=lis5[4];
                    for nn:=1 to 3 do
                     begin
                      N:=lis6[nn];
                      g[24+N]:=30;
                      g1[24+N]:=50;
                      lis7:=lis6;
                      lis7[nn]:=lis6[3];
                      for oo:=1 to 2 do
                       begin
                        O:=lis7[oo];
                        g[24+O]:=31;
                        g1[24+O]:=50;
                        lis8:=lis7;
                        lis8[oo]:=lis7[2];
                        P:=lis8[1];
                        g[24+P]:=32;
                        g1[24+P]:=20;
                        .....
if((g1[16+P]-20)*(g1[8+P]-20)*(g1[16+K]-5)*(g1[8+K]-5)<>0)then
 begin
  if((g1[I]-1)*(g1[8+I]-1)*(g1[8+J]-20)*(g1[16+J]-20)<>0)then
   begin
    if ((num-num1)*(num-num1)<0.5) then
     begin
      writeLn(datos2,'G1:',g[1],',',g[9],',',g[17],',',g[25]);
      writeLn(datos2,'G2:',g[2],',',g[10],',',g[18],',',g[26]);
      writeLn(datos2,'G3:',g[3],',',g[11],',',g[19],',',g[27]);
      writeLn(datos2,'G4:',g[4],',',g[12],',',g[20],',',g[28]);
      writeLn(datos2,'G5:',g[5],',',g[13],',',g[21],',',g[29]);
      writeLn(datos2,'G6:',g[6],',',g[14],',',g[22],',',g[30]);
      writeLn(datos2,'G7:',g[7],',',g[15],',',g[23],',',g[31]);
      writeLn(datos2,'G8:',g[8],',',g[16],',',g[24],',',g[32]);
     end;
    num:=num+1;
    .....P
                          end;
                        end;
                       end;
                      end;
                     end;
                    end;
                   end;
                  end;
                 end;
                end;
```

```
    end;
  end;
close(f);
close(f1);
close(f2);
end;


procedure bbB;
var
ii, I, jj, J, kk, K, ll, L, mm, M, nn, N, oo, O, pp, P:integer;
t,t1,c,c1,c2:integer;
var
lis1,lis2,lis3,lis4,lis5,lis6,lis7,lis8,lis: array[1..8] of integer;
f,f1,f2:text;
g,g1: array[1..32] of integer;
h: array[1..96] of integer;
h1: array[1..192] of integer;
h2: array[1..144] of integer;
begin
 assign(f, 'geo2226b.txt');
 assign(f1,'geo2226c1.txt');
 assign(f2,'geo2226c2.txt');
 reset(f);
 reset(f1);
 reset(f2);
 c:=0;
 c1:=0;
 c2:=0;
 while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
 while not Eoln(f1) do
  begin
   Inc(c1);
   Read(f1,h1[c1]);
  end;
 while not Eoln(f2) do
  begin
   Inc(c2);
   Read(f2,h2[c2]);
  end;
 for t:= 1 to 9 do
  begin
   case t of
    1 : begin
         for t1:=1 to 24 do g[t1]:=h[t1];
         for t1:=1 to 24 do g1[t1]:=h[t1+24];
        end;
    2 : begin
         for t1:=1 to 24 do g[t1]:=h[t1+48];
         for t1:=1 to 24 do g1[t1]:=h[t1+72];
        end;
    3 : begin
         for t1:=1 to 24 do g[t1]:=h1[t1];
         for t1:=1 to 24 do g1[t1]:=h1[t1+24];
        end;
    4 : begin
         for t1:=1 to 24 do g[t1]:=h1[t1+48];
         for t1:=1 to 24 do g1[t1]:=h1[t1+72];
        end;
```

```
  5 : begin
       for t1:=1 to 24 do g[t1]:=h1[t1+96];
       for t1:=1 to 24 do g1[t1]:=h1[t1+120];
      end;
  6 : begin
       for t1:=1 to 24 do g[t1]:=h1[t1+144];
       for t1:=1 to 24 do g1[t1]:=h1[t1+168];
      end;
  7 : begin
       for t1:=1 to 24 do g[t1]:=h2[t1];
       for t1:=1 to 24 do g1[t1]:=h2[t1+24];
      end;
  8 : begin
       for t1:=1 to 24 do g[t1]:=h2[t1+48];
       for t1:=1 to 24 do g1[t1]:=h2[t1+72];
      end;
  9 : begin
       for t1:=1 to 24 do g[t1]:=h2[t1+96];
       for t1:=1 to 24 do g1[t1]:=h2[t1+120];
      end;
end;
lis[1]:=1;
lis[2]:=2;
lis[3]:=3;
lis[4]:=4;
lis[5]:=5;
lis[6]:=6;
lis[7]:=7;
lis[8]:=8;
for ii:=1 to 8 do
 begin
  lis1:=lis;
  I:=lis1[ii];
  if(22>g[16+I]) then
   begin
    g[24+I]:=22;
    g1[24+I]:=20;
    lis2:=lis1;
    lis2[ii]:=lis1[8];
    for jj:=1 to 7 do
     begin
      J:=lis2[jj];
      if(25>g[16+J]) then
       begin
        g[24+J]:=25;
        g1[24+J]:=20;
        lis3:=lis2;
        lis3[jj]:=lis2[7];
        for kk:=1 to 6 do
         begin
          K:=lis3[kk];
          g[24+K]:=27;
          g1[24+K]:=5;
          lis4:=lis3;
          lis4[kk]:=lis3[6];
          for ll:=1 to 5 do
           begin
            L:=lis4[ll];
            g[24+L]:=28;
            g1[24+L]:=50;
            lis5:=lis4;
            lis5[ll]:=lis4[5];
            for mm:=1 to 4 do
```

```
                    begin
                     M:=lis5[mm];
                     g[24+M]:=29;
                     g1[24+M]:=50;
                     lis6:=lis5;
                     lis6[mm]:=lis5[4];
                     for nn:=1 to 3 do
                      begin
                       N:=lis6[nn];
                       g[24+N]:=30;
                       g1[24+N]:=50;
                       lis7:=lis6;
                       lis7[nn]:=lis6[3];
                       for oo:=1 to 2 do
                        begin
                         O:=lis7[oo];
                         g[24+O]:=31;
                         g1[24+O]:=50;
                         lis8:=lis7;
                         lis8[oo]:=lis7[2];
                         P:=lis8[1];
                         g[24+P]:=32;
                         g1[24+P]:=20;
                                .....
if((g1[16+P]-20)*(g1[8+P]-20)*(g1[16+K]-5)*(g1[8+K]-5)<>0)then
 begin
  if((g1[16+I]-20)*(g1[8+I]-20)*(g1[8+J]-20)*(g1[16+J]-20)<>0)then
   begin
    if ((num-num1)*(num-num1)<0.5) then
     begin
      writeLn(datos2,'G1:',g[1],',',g[9],',',g[17],',',g[25]);
      writeLn(datos2,'G2:',g[2],',',g[10],',',g[18],',',g[26]);
      writeLn(datos2,'G3:',g[3],',',g[11],',',g[19],',',g[27]);
      writeLn(datos2,'G4:',g[4],',',g[12],',',g[20],',',g[28]);
      writeLn(datos2,'G5:',g[5],',',g[13],',',g[21],',',g[29]);
      writeLn(datos2,'G6:',g[6],',',g[14],',',g[22],',',g[30]);
      writeLn(datos2,'G7:',g[7],',',g[15],',',g[23],',',g[31]);
      writeLn(datos2,'G8:',g[8],',',g[16],',',g[24],',',g[32]);
     end;
    num:=num+1;
    .....P
                           end;
                         end;
                        end;
                       end;
                      end;
                     end;
                    end;
                   end;
                  end;
                 end;
                end;
               end;
              end;
             end;
            end;
           end;
          end;
         end;
        end;
       end;
      end;
  end;
close(f);
close(f1);
close(f2);
end;



procedure bC;
var
ii, I, jj, J, kk, K, ll, L, mm, M, nn, N, oo, O, pp, P:integer;
```

```
t,t1,c,c1,c2,aux:integer;
var
lis1,lis2,lis3,lis4,lis5,lis6,lis7,lis8,lis: array[1..8] of integer;
f,f1,f2:text;
h: array[1..240] of integer;
h1: array[1..192] of integer;
h2: array[1..48] of integer;
g,g1: array[1..32] of integer;
begin
 assign(f,'geo2327b.txt');
 assign(f1,'geo2327c1.txt');
 assign(f2,'geo2327c2.txt');
 reset(f);
 reset(f1);
 reset(f2);
 c:=0;
 c1:=0;
 c2:=0;
 while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
 while not Eoln(f1) do
  begin
   Inc(c1);
   Read(f1,h1[c1]);
  end;
 while not Eoln(f2) do
  begin
   Inc(c2);
   Read(f2,h2[c2]);
  end;
 for t:= 1 to 10 do
  begin
   case t of
    1 : begin
         for t1:=1 to 24 do g[t1]:=h[t1];
         for t1:=1 to 24 do g1[t1]:=h[t1+24];
        end;
    2 : begin
         for t1:=1 to 24 do g[t1]:=h[t1+48];
         for t1:=1 to 24 do g1[t1]:=h[t1+72];
        end;
    3 : begin
         for t1:=1 to 24 do g[t1]:=h[t1+96];
         for t1:=1 to 24 do g1[t1]:=h[t1+120];
        end;
    4 : begin
         for t1:=1 to 24 do g[t1]:=h[t1+144];
         for t1:=1 to 24 do g1[t1]:=h[t1+168];
        end;
    5 : begin
         for t1:=1 to 24 do g[t1]:=h[t1+192];
         for t1:=1 to 24 do g1[t1]:=h[t1+216];
        end;
    6 : begin
         for t1:=1 to 24 do g[t1]:=h1[t1];
         for t1:=1 to 24 do g1[t1]:=h1[t1+24];
        end;
    7 : begin
         for t1:=1 to 24 do g[t1]:=h1[t1+48];
         for t1:=1 to 24 do g1[t1]:=h1[t1+72];
```

```
      end;
  8 : begin
       for t1:=1 to 24 do g[t1]:=h1[t1+96];
       for t1:=1 to 24 do g1[t1]:=h1[t1+120];
      end;
  9 : begin
       for t1:=1 to 24 do g[t1]:=h1[t1+144];
       for t1:=1 to 24 do g1[t1]:=h1[t1+168];
      end;
  10 : begin
       for t1:=1 to 24 do g[t1]:=h2[t1];
       for t1:=1 to 24 do g1[t1]:=h2[t1+24];
      end;

end;
lis[1]:=1;
lis[2]:=2;
lis[3]:=3;
lis[4]:=4;
lis[5]:=5;
lis[6]:=6;
lis[7]:=7;
lis[8]:=8;
for ii:=1 to 8 do
 begin
  lis1:=lis;
  I:=lis1[ii];
  if(23>g[16+I]) then
   begin
    g[24+I]:=23;
    g1[24+I]:=5;
    lis2:=lis1;
    lis2[ii]:=lis1[8];
    for jj:=1 to 7 do
     begin
      J:=lis2[jj];
      if(25>g[16+J]) then
       begin
        g[24+J]:=25;
        g1[24+J]:=20;
        lis3:=lis2;
        lis3[jj]:=lis2[7];
        for kk:=1 to 6 do
         begin
          K:=lis3[kk];
          g[24+K]:=26;
          g1[24+K]:=20;
          lis4:=lis3;
          lis4[kk]:=lis3[6];
          for ll:=1 to 5 do
           begin
            L:=lis4[ll];
            g[24+L]:=28;
            g1[24+L]:=50;
            lis5:=lis4;
            lis5[ll]:=lis4[5];
            for mm:=1 to 4 do
             begin
              M:=lis5[mm];
              g[24+M]:=29;
              g1[24+M]:=50;
              lis6:=lis5;
              lis6[mm]:=lis5[4];
```

```
                    for nn:=1 to 3 do
                     begin
                      N:=lis6[nn];
                      g[24+N]:=30;
                      g1[24+N]:=50;
                      lis7:=lis6;
                      lis7[nn]:=lis6[3];
                      for oo:=1 to 2 do
                       begin
                        O:=lis7[oo];
                        g[24+O]:=31;
                        g1[24+O]:=50;
                        lis8:=lis7;
                        lis8[oo]:=lis7[2];
                        P:=lis8[1];
                        g[24+P]:=32;
                        g1[24+P]:=20;
                        aux:=(g1[16+P]-20)*(g1[8+P]-20)*(g1[16+K]-20);
                        .....
if(aux*(g1[8+K]-20)*(g[16+K]-27)<>0)then
 begin
  if((g1[I+8]-5)*(g1[16+I]-5)*(g1[8+J]-20)*(g1[16+J]-20)<>0)then
   begin
    if ((num-num1)*(num-num1)<0.5) then
     begin
      writeLn(datos2,'G1:',g[1],',',g[9],',',g[17],',',g[25]);
      writeLn(datos2,'G2:',g[2],',',g[10],',',g[18],',',g[26]);
      writeLn(datos2,'G3:',g[3],',',g[11],',',g[19],',',g[27]);
      writeLn(datos2,'G4:',g[4],',',g[12],',',g[20],',',g[28]);
      writeLn(datos2,'G5:',g[5],',',g[13],',',g[21],',',g[29]);
      writeLn(datos2,'G6:',g[6],',',g[14],',',g[22],',',g[30]);
      writeLn(datos2,'G7:',g[7],',',g[15],',',g[23],',',g[31]);
      writeLn(datos2,'G8:',g[8],',',g[16],',',g[24],',',g[32]);
     end;
    num:=num+1;
    .....P
                          end;
                        end;
                       end;
                      end;
                     end;
                    end;
                   end;
                  end;
                 end;
                end;
               end;
              end;
             end;
            end;
           end;
          end;
         end;
        end;
       end;
      end;
     end;
    end;
  end;
close(f);
close(f1);
close(f2);
end;


procedure bD;

var
ii, I, jj, J, kk, K, ll, L, mm, M, nn, N, oo, O, pp, P:integer;
t,t1,c,c1,aux,aux1:integer;
var
lis1,lis2,lis3,lis4,lis5,lis6,lis7,lis8,lis: array[1..8] of integer;
f,f1:text;
g,g1: array[1..32] of integer;
```

```
h: array[1..48] of integer;
h1: array[1..96] of integer;
begin
 assign(f,'geo2428b.txt');
 assign(f1,'geo2428c1.txt');
 reset(f);
 reset(f1);
 c:=0;
 c1:=0;
 while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
 while not Eoln(f1) do
  begin
   Inc(c1);
   Read(f1,h1[c1]);
  end;
 for t:= 1 to 3 do
  begin
   case t of
    1 : begin
         for t1:=1 to 24 do g[t1]:=h[t1];
         for t1:=1 to 24 do g1[t1]:=h[t1+24];
        end;
    2 : begin
         for t1:=1 to 24 do g[t1]:=h1[t1];
         for t1:=1 to 24 do g1[t1]:=h1[t1+24];
        end;
    3 : begin
         for t1:=1 to 24 do g[t1]:=h1[t1+48];
         for t1:=1 to 24 do g1[t1]:=h1[t1+72];
        end;
   end;
   lis[1]:=1;
   lis[2]:=2;
   lis[3]:=3;
   lis[4]:=4;
   lis[5]:=5;
   lis[6]:=6;
   lis[7]:=7;
   lis[8]:=8;
   for ii:=1 to 8 do
    begin
     lis1:=lis;
     I:=lis1[ii];
     if(24>g[16+I]) then
      begin
       g[24+I]:=24;
       g1[24+I]:=5;
       lis2:=lis1;
       lis2[ii]:=lis1[8];
       for jj:=1 to 7 do
        begin
         J:=lis2[jj];
         if(25>g[16+J]) then
          begin
           g[24+J]:=25;
           g1[24+J]:=20;
           lis3:=lis2;
           lis3[jj]:=lis2[7];
           for kk:=1 to 6 do
```

```
              begin
               K:=lis3[kk];
               g[24+K]:=26;
               g1[24+K]:=20;
               lis4:=lis3;
               lis4[kk]:=lis3[6];
               for ll:=1 to 5 do
                begin
                 L:=lis4[ll];
                 g[24+L]:=27;
                 g1[24+L]:=5;
                 lis5:=lis4;
                 lis5[ll]:=lis4[5];
                 for mm:=1 to 4 do
                  begin
                   M:=lis5[mm];
                   g[24+M]:=29;
                   g1[24+M]:=50;
                   lis6:=lis5;
                   lis6[mm]:=lis5[4];
                   for nn:=1 to 3 do
                    begin
                     N:=lis6[nn];
                     g[24+N]:=30;
                     g1[24+N]:=50;
                     lis7:=lis6;
                     lis7[nn]:=lis6[3];
                     for oo:=1 to 2 do
                      begin
                       O:=lis7[oo];
                       g[24+O]:=31;
                       g1[24+O]:=50;
                       lis8:=lis7;
                       lis8[oo]:=lis7[2];
                       P:=lis8[1];
                       g[24+P]:=32;
                       g1[24+P]:=20;
                       aux:=(g1[16+P]-20)*(g1[8+P]-20)*(g1[16+O]-50);
                       aux1:=(g[L+16]-28)*(g1[16+L]-5)*(g1[8+L]-5);
                       .....
if(aux*(g1[16+N]-50)*(g1[16+M]-50)<>0)then
 begin
  if(aux1*(g[16+K]-28)*(g1[8+K]-20)*(g1[16+K]-20)<>0)then
   begin
    if((g1[16+J]-20)*(g1[8+J]-20)*(g1[8+I]-5)*(g1[16+I]-5)<>0)then
     begin
      if ((num-num1)*(num-num1)<0.5) then
       begin
        writeLn(datos2,'G1:',g[1],',',g[9],',',g[17],',',g[25]);
        writeLn(datos2,'G2:',g[2],',',g[10],',',g[18],',',g[26]);
        writeLn(datos2,'G3:',g[3],',',g[11],',',g[19],',',g[27]);
        writeLn(datos2,'G4:',g[4],',',g[12],',',g[20],',',g[28]);
        writeLn(datos2,'G5:',g[5],',',g[13],',',g[21],',',g[29]);
        writeLn(datos2,'G6:',g[6],',',g[14],',',g[22],',',g[30]);
        writeLn(datos2,'G7:',g[7],',',g[15],',',g[23],',',g[31]);
        writeLn(datos2,'G8:',g[8],',',g[16],',',g[24],',',g[32]);
       end;
      num:=num+1;
      .....P
                              end;
                            end;
                          end;
                        end;
```

```
                    end;
                  end;
                end;
              end;
            end;
          end;
        end;
      end;
  end;
close(f);
close(f1);
end;

procedure bE;
var
ii, I, jj, J, kk, K, ll, L, mm, M, nn, N, oo, O, pp, P:integer;
t,t1,c1,c2,c:integer;
var
lis1,lis2,lis3,lis4,lis5,lis6,lis7,lis8,lis: array[1..8] of integer;
f,f1,f2:text;
h:array[1..192] of integer;
h1:array[1..144] of integer;
h2:array[1..48] of integer;
g,g1: array[1..32] of integer;
begin
 assign(f,'geo23242526b.txt');
 assign(f1,'geo23242526c1.txt');
 assign(f2,'geo23242526c2.txt');
 reset(f);
 reset(f1);
 reset(f2);
 c:=0;
 c1:=0;
 c2:=0;
 while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
 while not Eoln(f1) do
  begin
   Inc(c1);
   Read(f1,h1[c1]);
  end;
 while not Eoln(f2) do
  begin
   Inc(c2);
   Read(f2,h2[c2]);
  end;
 for t:= 1 to 8 do
  begin
   case t of
    1 : begin
         for t1:=1 to 24 do g[t1]:=h[t1];
         for t1:=1 to 24 do g1[t1]:=h[t1+24];
        end;
    2 : begin
         for t1:=1 to 24 do g[t1]:=h[t1+48];
         for t1:=1 to 24 do g1[t1]:=h[t1+72];
        end;
    3 : begin
         for t1:=1 to 24 do g[t1]:=h[t1+96];
         for t1:=1 to 24 do g1[t1]:=h[t1+120];
```

```
      end;
  4 : begin
       for t1:=1 to 24 do g[t1]:=h[t1+144];
       for t1:=1 to 24 do g1[t1]:=h[t1+168];
      end;
  5 : begin
       for t1:=1 to 24 do g[t1]:=h1[t1];
       for t1:=1 to 24 do g1[t1]:=h1[t1+24];
      end;
  6 : begin
       for t1:=1 to 24 do g[t1]:=h1[t1+48];
       for t1:=1 to 24 do g1[t1]:=h1[t1+72];
      end;
  7 : begin
       for t1:=1 to 24 do g[t1]:=h1[t1+96];
       for t1:=1 to 24 do g1[t1]:=h1[t1+120];
      end;
  8 : begin
       for t1:=1 to 24 do g[t1]:=h2[t1];
       for t1:=1 to 24 do g1[t1]:=h2[t1+24];
      end;
 end;
 lis[1]:=1;
 lis[2]:=2;
 lis[3]:=3;
 lis[4]:=4;
 lis[5]:=5;
 lis[6]:=6;
 lis[7]:=7;
 lis[8]:=8;
 for ii:=1 to 8 do
  begin
   lis1:=lis;
   I:=lis1[ii];
   if(23>g[16+I]) then
    begin
     g[24+I]:=23;
     g1[24+I]:=5;
     lis2:=lis1;
     lis2[ii]:=lis1[8];
     for jj:=1 to 7 do
      begin
       J:=lis2[jj];
       if(24>g[16+J]) then
        begin
         g[24+J]:=24;
         g1[24+J]:=5;
         lis3:=lis2;
         lis3[jj]:=lis2[7];
         for kk:=1 to 6 do
          begin
           K:=lis3[kk];
           g[24+K]:=27;
           g1[24+K]:=5;
           lis4:=lis3;
           lis4[kk]:=lis3[6];
           for ll:=1 to 5 do
            begin
             L:=lis4[ll];
             g[24+L]:=28;
             g1[24+L]:=50;
             lis5:=lis4;
             lis5[ll]:=lis4[5];
```

```
                    for mm:=1 to 4 do
                     begin
                      M:=lis5[mm];
                      g[24+M]:=29;
                      g1[24+M]:=50;
                      lis6:=lis5;
                      lis6[mm]:=lis5[4];
                      for nn:=1 to 3 do
                       begin
                        N:=lis6[nn];
                        g[24+N]:=30;
                        g1[24+N]:=50;
                        lis7:=lis6;
                        lis7[nn]:=lis6[3];
                        for oo:=1 to 2 do
                         begin
                          O:=lis7[oo];
                          g[24+O]:=31;
                          g1[24+O]:=50;
                          lis8:=lis7;
                          lis8[oo]:=lis7[2];
                          P:=lis8[1];
                          g[24+P]:=32;
                          g1[24+P]:=20;
                          .....
if((g1[16+P]-20)*(g1[8+P]-20)*(g1[16+K]-5)*(g1[8+K]-5)<>0)then
 begin
  if((g1[16+I]-5)*(g1[8+I]-5)*(g1[8+J]-5)*(g1[16+J]-5)<>0)then
   begin
    if ((num-num1)*(num-num1)<0.5) then
     begin
      writeLn(datos2,'G1:',g[1],',',g[9],',',g[17],',',g[25]);
      writeLn(datos2,'G2:',g[2],',',g[10],',',g[18],',',g[26]);
      writeLn(datos2,'G3:',g[3],',',g[11],',',g[19],',',g[27]);
      writeLn(datos2,'G4:',g[4],',',g[12],',',g[20],',',g[28]);
      writeLn(datos2,'G5:',g[5],',',g[13],',',g[21],',',g[29]);
      writeLn(datos2,'G6:',g[6],',',g[14],',',g[22],',',g[30]);
      writeLn(datos2,'G7:',g[7],',',g[15],',',g[23],',',g[31]);
      writeLn(datos2,'G8:',g[8],',',g[16],',',g[24],',',g[32]);
     end;
    num:=num+1;
    .....P
                         end;
                        end;
                       end;
                      end;
                     end;
                    end;
                   end;
                  end;
                 end;
                end;
               end;
              end;
             end;
            end;
           end;
          end;
         end;
        end;
       end;
      end;
     end;
    end;
  end;
close(f);
close(f1);
close(f2);
end;


begin

assign(datos2,'sorteoperfecto.txt');
```

```
rewrite(datos2);
randomize;
for con:=1 to 10 do
begin
 num1:=random(173160) +1;
 num:=1;
 bA;
 bbB;
 bC;
 bD;
 bE;
end;

close(datos2);

end.
```

B.5. Sorteo 323

Sorteo 323. Todos los archivos usados .txt son las distintas soluciones geográficas de los distintos sorteos.

```
program sorteo323;
var
lis1,lis2,lis3:array[1..3]of integer;
liss1,liss2,liss3:array[1..2]of integer;
lisss1, lisss2,lisss3:array[1..3]of integer;
alea,alea2:integer;
c23:integer;
datos:text;


procedure sorteo4;
var
h:array[1..54] of integer;
g1:array[1..9] of integer;
g:array[1..12] of integer;
f:text;
t,t1,t2,t3,t4,t5,t6,t7:integer;
r,r1,r2,r3,r4,r5,r6,c:integer;
lista1,lista2,lista3:array[1..3]of integer;
vec, vec1,vec2,vec3:array[1..3]of integer;

begin
assign(f,'Ogeo1011A.txt');
reset(f);
c:=0;
while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
r:=1 + random(6);
for t:=1 to 9 do g1[t]:=h[t+(r-1)*9];
lista1:=lis1;
lista2:=lis2;
lista3:=lis3;
for t:=1 to 3 do vec[t]:=t;
g[1]:=1;
g[2]:=2;
g[3]:=3;
r1:=random(3)+1;
g[3+r1]:=lista1[1];
vec1:=vec;
vec1[r1]:=vec1[3];
r2:=random(2)+1;
g[3+vec1[r2]]:=lista1[2];
vec1[r2]:=vec1[2];
g[3+vec1[1]]:=lista1[3];

for t:=1 to 3 do
 begin
  if (g1[3+t] = 0) then
   begin
    g[6+t]:=lista2[1];
    for t1:=1 to 2 do
     begin
      vec1:=vec;
      vec1[t]:=vec1[3];
```

```
       if(g1[3+vec1[t1]]=1)then
        begin
         g[6+vec1[t1]]:=lista2[2];
         vec1[t1]:=vec1[2];
         g[6+vec1[1]]:=lista2[3];
        end;
      end;
    end;
 end;
for t2:=1 to 3 do
 begin
  if(g1[6+t2]=5) then
   begin
    vec2:=vec;
    g[9+t2]:=lista3[1];
    vec2[t2]:=vec2[3];
    r3:=random(2)+1;
    g[9+vec2[r3]]:=lista3[2];
    vec2[r3]:=vec2[2];
    g[9+vec2[1]]:=lista3[3];
   end;
 end;
writeLn(datos,g[1],'␣',g[4],'␣',g[7],'␣',g[10]);
writeLn(datos,g[2],'␣',g[5],'␣',g[8],'␣',g[11]);
writeLn(datos,g[3],'␣',g[6],'␣',g[9],'␣',g[12]);

close(f);
end;


procedure sorteo1;
var
h:array[1..18] of integer;
g1:array[1..9] of integer;
g:array[1..12] of integer;
f:text;
t,t1,t2,t3,t4,t5,t6,t7:integer;
r,r1,r2,r3,r4,r5,r6,c:integer;
lista1,lista2,lista3:array[1..3]of integer;
vec, vec1,vec2,vec3:array[1..3]of integer;

begin
assign(f,'Oinicial1A.txt');
reset(f);
c:=0;
while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
r:=1 + random(2);
for t:=1 to 9 do g1[t]:=h[t+(r-1)*9];
lista1:=lis1;
lista2:=lis2;
lista3:=lis3;
for t:=1 to 3 do vec[t]:=t;
g[1]:=1;
g[2]:=2;
g[3]:=3;
r1:=random(3)+1;
g[3+r1]:=lista1[1];
vec1:=vec;
vec1[r1]:=vec1[3];
```

```
r2:=random(2)+1;
g[3+vec1[r2]]:=lista1[2];
vec1[r2]:=vec1[2];
g[3+vec1[1]]:=lista1[3];

for t:=1 to 3 do
 begin
  if (g1[3+t] = 0) then
   begin
    g[6+t]:=lista2[1];
    for t1:=1 to 2 do
     begin
      vec1:=vec;
      vec1[t]:=vec1[3];
      if(g1[3+vec1[t1]]=1)then
       begin
        g[6+vec1[t1]]:=lista2[2];
        vec1[t1]:=vec1[2];
        g[6+vec1[1]]:=lista2[3];
       end;
     end;
   end;
 end;
for t2:=1 to 3 do
 begin
  if(g1[6+t2]=5) then
   begin
    vec2:=vec;
    g[9+t2]:=lista3[1];
    vec2[t2]:=vec2[3];
    r3:=random(2)+1;
    g[9+vec2[r3]]:=lista3[2];
    vec2[r3]:=vec2[2];
    g[9+vec2[1]]:=lista3[3];
   end;
 end;
writeLn(datos,g[1],'␣',g[4],'␣',g[7],'␣',g[10]);
writeLn(datos,g[2],'␣',g[5],'␣',g[8],'␣',g[11]);
writeLn(datos,g[3],'␣',g[6],'␣',g[9],'␣',g[12]);

close(f);
end;

procedure sorteo2;
var
lista1,lista2,lista3:array[1..2] of integer;
vec,vec1,vec2,vec3:array[1..2] of integer;
t,t1,t2,r,r1,r2,r3:integer;
i:integer;

g:array[1..8] of integer;

begin
g[1]:=4;
g[2]:=5;
lista1:=liss1;
lista2:=liss2;
lista3:=liss3;
vec[1]:=1;
vec[2]:=2;
r:=random(2)+1;
vec1:=vec;
g[2+vec1[r]]:=lista1[1];
```

```
vec1[r]:=vec1[2];
g[2+vec1[1]]:=lista1[2];

r1:=random(2)+1;
vec2:=vec;
g[4+vec2[r1]]:=lista2[1];
vec2[r1]:=vec2[2];
g[4+vec2[1]]:=lista2[2];

r2:=random(2)+1;
vec3:=vec;
g[6+vec3[r2]]:=lista3[1];
vec3[r2]:=vec3[2];
g[6+vec3[1]]:=lista3[2];

writeLn(datos,g[1],'␣',g[3],'␣',g[5],'␣',g[7]);
writeLn(datos,g[2],'␣',g[4],'␣',g[6],'␣',g[8]);

end;

procedure sorteo3;
var
h:array[1..144]of integer;
g1:array[1..9]of integer;
g:array[1..12]of integer;
f:text;
t,t1,t2,t3,r,r1,r2,r3,c:integer;
lista1,lista2,lista3:array[1..3]of integer;
vec,vec1,vec2,vec3:array[1..3]of integer;

begin
assign(f,'Oinicial1B.txt');
reset(f);
c:=0;
while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
lista1:=lisss1;
lista2:=lisss2;
lista3:=lisss3;
r:=1+random(16);
for t:=1 to 9 do g1[t]:=h[t+(r-1)*9];
g[1]:=6;
g[2]:=7;
g[3]:=8;
for t:=1 to 3 do vec[t]:=t;
for t:=1 to 3 do
 begin
  if(g1[t]=0)then
   begin
    g[3+t]:=lista1[1];
    for t1:=1 to 2 do
     begin
      vec1:=vec;
      vec1[t]:=vec1[3];
      if (g1[vec1[t1]]=1)then
       begin
        g[3+vec1[t1]]:=lista1[2];
        vec1[t1]:=vec1[2];
        g[3+vec1[1]]:=lista1[3];
       end;
```

```
      end;
    end;
 end;
for t2:=1 to 3 do
 begin
  if(g1[3+t2]=20) then
   begin
    g[6+t2]:=lista2[1];
    vec2:=vec;
    vec2[t2]:=vec2[3];
    r1:=random(2)+1;
    g[6+vec2[r1]]:=lista2[2];
    vec2[r1]:=vec2[2];
    g[6+vec2[1]]:=lista2[3];
   end;
 end;
for t3:=1 to 3 do
 begin
  if(g1[6+t3]=20) then
   begin
    g[9+t3]:=lista3[1];
    vec3:=vec;
    vec3[t3]:=vec3[3];
    r2:=random(2)+1;
    g[9+vec3[r2]]:=lista3[2];
    vec3[r2]:=vec3[2];
    g[9+vec3[1]]:=lista3[3];
   end;
 end;
writeLn(datos,g[1],'␣',g[4],'␣',g[7],'␣',g[10]);
writeLn(datos,g[2],'␣',g[5],'␣',g[8],'␣',g[11]);
writeLn(datos,g[3],'␣',g[6],'␣',g[9],'␣',g[12]);
close(f);
end;

procedure sorteo5;
var
f:text;
h:array[1..54]of integer;
g1:array[1..9]of integer;
g:array[1..12]of integer;
lista1,lista2,lista3:array[1..3] of integer;
vec,vec1,vec2,vec3,vec4:array[1..3] of integer;
r,r1,r2,r3,t,t1,t2,t3,t4,c:integer;

begin
assign(f,'Ogeo1413A.txt');
reset(f);
c:=0;
while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
lista1:=lis1;
lista2:=lis2;
lista3:=lis3;
r:=1+random(6);
for t:=1 to 9 do g1[t]:=h[t+(r-1)*9];
g[1]:=1;
g[2]:=2;
g[3]:=3;
for t:=1 to 3 do vec[t]:=t;
```

```
for t:=1 to 3 do
 begin
  if(g1[t]=5)then
   begin
    g[3+t]:=lista1[1];
    vec1:=vec;
    vec1[t]:=vec1[3];
    r1:=1+random(2);
    g[3+vec1[r1]]:=lista1[2];
    vec1[r1]:=vec1[2];
    g[3+vec1[1]]:=lista1[3];
   end;
 end;
for t1:=1 to 3 do
 begin
  if (g1[3+t1] = 0) then
   begin
    g[6+t1]:=lista2[1];
    for t2:=1 to 2 do
     begin
      vec1:=vec;
      vec1[t1]:=vec1[3];
      if(g1[3+vec1[t2]]=1)then
       begin
        g[6+vec1[t2]]:=lista2[2];
        vec1[t2]:=vec1[2];
        g[6+vec1[1]]:=lista2[3];
       end;
     end;
   end;
 end;
for t2:=1 to 3 do
 begin
  if(g1[6+t2]=5) then
   begin
    vec2:=vec;
    g[9+t2]:=lista3[1];
    vec2[t2]:=vec2[3];
    r3:=random(2)+1;
    g[9+vec2[r3]]:=lista3[2];
    vec2[r3]:=vec2[2];
    g[9+vec2[1]]:=lista3[3];
   end;
 end;
writeLn(datos,g[1],'␣',g[4],'␣',g[7],'␣',g[10]);
writeLn(datos,g[2],'␣',g[5],'␣',g[8],'␣',g[11]);
writeLn(datos,g[3],'␣',g[6],'␣',g[9],'␣',g[12]);
close(f);
end;

procedure sorteo6;
var
f:text;
h:array[1..36]of integer;
g1:array[1..9]of integer;
g:array[1..12]of integer;
lista1,lista2,lista3:array[1..3] of integer;
vec,vec1,vec2,vec3,vec4:array[1..3] of integer;
r,r1,r2,r3,t,t1,t2,t3,t4,c:integer;

begin
assign(f,'Ogeo1413B.txt');
reset(f);
```

```
c:=0;
while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
lista1:=lisss1;
lista2:=lisss2;
lista3:=lisss3;
r:=1+random(4);
for t:=1 to 9 do g1[t]:=h[t+(r-1)*9];
g[1]:=6;
g[2]:=7;
g[3]:=8;
for t:=1 to 3 do vec[t]:=t;
for t:=1 to 3 do
 begin
  if(g1[t]=1)then
   begin
    g[3+t]:=lista1[1];
    vec1:=vec;
    vec1[t]:=vec1[3];
    r1:=1+random(2);
    g[3+vec1[r1]]:=lista1[2];
    vec1[r1]:=vec1[2];
    g[3+vec1[1]]:=lista1[3];
   end;
 end;
for t:=1 to 3 do
 begin
  if(g1[t+3]=20)then
   begin
    g[6+t]:=lista2[1];
    vec1:=vec;
    vec1[t]:=vec1[3];
    r1:=1+random(2);
    g[6+vec1[r1]]:=lista2[2];
    vec1[r1]:=vec1[2];
    g[6+vec1[1]]:=lista2[3];
   end;
 end;
for t:=1 to 3 do
 begin
  if(g1[t+6]=20)then
   begin
    g[9+t]:=lista3[1];
    vec1:=vec;
    vec1[t]:=vec1[3];
    r1:=1+random(2);
    g[9+vec1[r1]]:=lista3[2];
    vec1[r1]:=vec1[2];
    g[9+vec1[1]]:=lista3[3];
   end;
 end;

writeLn(datos,g[1],'␣',g[4],'␣',g[7],'␣',g[10]);
writeLn(datos,g[2],'␣',g[5],'␣',g[8],'␣',g[11]);
writeLn(datos,g[3],'␣',g[6],'␣',g[9],'␣',g[12]);
close(f);
end;

procedure sorteo7;
var
```

```
f:text;
h:array[1..36]of integer;
g1:array[1..9]of integer;
g:array[1..12]of integer;
lista1,lista2,lista3:array[1..3] of integer;
vec,vec1,vec2,vec3,vec4:array[1..3] of integer;
r,r1,r2,r3,t,t1,t2,t3,t4,c:integer;

begin
assign(f,'Ogeo2223A.txt');
reset(f);
c:=0;
while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
lista1:=lis1;
lista2:=lis2;
lista3:=lis3;
r:=1+random(4);
for t:=1 to 9 do g1[t]:=h[t+(r-1)*9];
g[1]:=1;
g[2]:=2;
g[3]:=3;
for t:=1 to 3 do vec[t]:=t;
r1:=random(3)+1;
g[3+r1]:=lista1[1];
vec1:=vec;
vec1[r1]:=vec1[3];
r2:=random(2)+1;
g[3+vec1[r2]]:=lista1[2];
vec1[r2]:=vec1[2];
g[3+vec1[1]]:=lista1[3];
for t1:=1 to 3 do
 begin
  if (g1[3+t1] = 0) then
   begin
    g[6+t1]:=lista2[1];
    for t2:=1 to 2 do
     begin
      vec1:=vec;
      vec1[t1]:=vec1[3];
      if(g1[3+vec1[t2]]=1)then
       begin
        g[6+vec1[t2]]:=lista2[2];
        vec1[t2]:=vec1[2];
        g[6+vec1[1]]:=lista2[3];
       end;
     end;
   end;
 end;
for t2:=1 to 3 do
 begin
  if(g1[6+t2]=5) then
   begin
    vec2:=vec;
    g[9+t2]:=lista3[1];
    vec2[t2]:=vec2[3];
    r3:=random(2)+1;
    g[9+vec2[r3]]:=lista3[2];
    vec2[r3]:=vec2[2];
    g[9+vec2[1]]:=lista3[3];
```

```
    end;
 end;
writeLn(datos,g[1],'␣',g[4],'␣',g[7],'␣',g[10]);
writeLn(datos,g[2],'␣',g[5],'␣',g[8],'␣',g[11]);
writeLn(datos,g[3],'␣',g[6],'␣',g[9],'␣',g[12]);
close(f);
end;

procedure sorteo8;
var
f:text;
h:array[1..12]of integer;
g1:array[1..6]of integer;
g:array[1..8]of integer;
lista1,lista2,lista3:array[1..2] of integer;
vec,vec1,vec2,vec3,vec4:array[1..2] of integer;
r,r1,r2,r3,t,t1,t2,t3,t4,c:integer;

begin
assign(f,'Ogeo2223M.txt');
reset(f);
c:=0;
while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
lista1:=liss1;
lista2:=liss2;
lista3:=liss3;
r:=1+random(2);
for t:=1 to 6 do g1[t]:=h[t+(r-1)*6];
g[1]:=4;
g[2]:=5;
for t:=1 to 2 do vec[t]:=t;
r:=random(2)+1;
vec1:=vec;
g[2+vec1[r]]:=lista1[1];
vec1[r]:=vec1[2];
g[2+vec1[1]]:=lista1[2];
for t:=1 to 2 do
 begin
  if(g1[2+t]=20)then
   begin
    g[4+t]:=lista2[1];
    vec2:=vec;
    vec2[t]:=vec2[2];
    g[4+vec2[1]]:=lista2[2];
   end;
 end;
for t:=1 to 2 do
 begin
  if(g1[4+t]=50)then
   begin
    g[6+t]:=lista3[1];
    vec2:=vec;
    vec2[t]:=vec2[2];
    g[6+vec2[1]]:=lista3[2];
   end;
 end;

writeLn(datos,g[1],'␣',g[3],'␣',g[5],'␣',g[7]);
writeLn(datos,g[2],'␣',g[4],'␣',g[6],'␣',g[8]);
```

```
close(f);
end;

procedure sorteo9;
var
f:text;
h:array[1..72]of integer;
g1:array[1..9]of integer;
g:array[1..12]of integer;
lista1,lista2,lista3:array[1..3] of integer;
vec,vec1,vec2,vec3,vec4:array[1..3] of integer;
r,r1,r2,r3,t,t1,t2,t3,t4,c:integer;

begin
assign(f,'Ogeo2223B.txt');
reset(f);
c:=0;
while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
lista1:=lisss1;
lista2:=lisss2;
lista3:=lisss3;
r:=1+random(8);
for t:=1 to 9 do g1[t]:=h[t+(r-1)*9];
g[1]:=6;
g[2]:=7;
g[3]:=8;
for t:=1 to 3 do vec[t]:=t;
for t:=1 to 3 do
 begin
  if(g1[t]=0)then
   begin
    g[3+t]:=lista1[1];
    for t1:=1 to 2 do
     begin
      vec1:=vec;
      vec1[t]:=vec1[3];
      if (g1[vec1[t1]]=1)then
       begin
        g[3+vec1[t1]]:=lista1[2];
        vec1[t1]:=vec1[2];
        g[3+vec1[1]]:=lista1[3];
       end;
     end;
   end;
 end;
for t2:=1 to 3 do
 begin
  if(g1[3+t2]=20) then
   begin
    g[6+t2]:=lista2[1];
    vec2:=vec;
    vec2[t2]:=vec2[3];
    r1:=random(2)+1;
    g[6+vec2[r1]]:=lista2[2];
    vec2[r1]:=vec2[2];
    g[6+vec2[1]]:=lista2[3];
   end;
 end;
r2:=random(3)+1;
```

```
vec3:=vec;
g[9+r2]:=lista3[1];
vec3[r2]:=vec3[3];
r3:=random(2)+1;
g[9+vec3[r3]]:=lista3[2];
vec3[r3]:=vec3[2];
g[9+vec3[1]]:=lista3[3];

writeLn(datos,g[1],'␣',g[4],'␣',g[7],'␣',g[10]);
writeLn(datos,g[2],'␣',g[5],'␣',g[8],'␣',g[11]);
writeLn(datos,g[3],'␣',g[6],'␣',g[9],'␣',g[12]);
close(f);
end;

procedure sorteo10;
var
f:text;
h:array[1..36]of integer;
g1:array[1..9]of integer;
g:array[1..12]of integer;
lista1,lista2,lista3:array[1..3] of integer;
vec,vec1,vec2,vec3,vec4:array[1..3] of integer;
r,r1,r2,r3,t,t1,t2,t3,t4,c:integer;

begin
assign(f,'Ogeo2024A.txt');
reset(f);
c:=0;
while not Eoln(f) do
 begin
  Inc(c);
  Read(f,h[c]);
 end;
lista1:=lis1;
lista2:=lis2;
lista3:=lis3;
r:=1+random(4);
for t:=1 to 9 do g1[t]:=h[t+(r-1)*9];
g[1]:=1;
g[2]:=2;
g[3]:=3;
for t:=1 to 3 do vec[t]:=t;
r1:=random(3)+1;
g[4]:=lista1[r1];
lista1[r1]:=lista1[3];
r2:=random(2)+1;
g[5]:=lista1[r2];
lista1[r2]:=lista1[2];
g[6]:=lista1[1];

for t1:=1 to 3 do
 begin
  if (g1[3+t1] = 1) then
   begin
    g[6+t1]:=lista2[1];
    for t2:=1 to 2 do
     begin
      vec1:=vec;
      vec1[t1]:=vec1[3];
      if(g1[3+vec1[t2]]=20)then
       begin
        g[6+vec1[t2]]:=lista2[2];
        vec1[t2]:=vec1[2];
```

```
          g[6+vec1[1]]:=lista2[3];
        end;
      end;
    end;
 end;
for t2:=1 to 3 do
 begin
  if(g1[6+t2]=5) then
   begin
    vec2:=vec;
    g[9+t2]:=lista3[1];
    vec2[t2]:=vec2[3];
    r3:=random(2)+1;
    g[9+vec2[r3]]:=lista3[2];
    vec2[r3]:=vec2[2];
    g[9+vec2[1]]:=lista3[3];
   end;
 end;
writeLn(datos,g[1],'␣',g[4],'␣',g[7],'␣',g[10]);
writeLn(datos,g[2],'␣',g[5],'␣',g[8],'␣',g[11]);
writeLn(datos,g[3],'␣',g[6],'␣',g[9],'␣',g[12]);
close(f);
end;


begin
 assign(datos,'sorteo323.txt');
 rewrite(datos);
 randomize;
 begin
 alea:=random(20)+1;
 case alea of
  1..2 :begin
                for c23:=1 to 3 do
                 begin
                  lis1[c23]:=13+c23;
                  lis2[c23]:=19+c23;
                  lis3[c23]:=28-c23;
                  lisss1[c23]:=10+c23;
                  lisss2[c23]:=16+c23;
                  lisss3[c23]:=33-c23;
                 end;
                for c23:=1 to 2 do liss1[c23]:=8+c23;
                for c23:=1 to 2 do liss2[c23]:=22+c23;
                for c23:=1 to 2 do liss3[c23]:=27+c23;
                sorteo1;
                sorteo2;
                sorteo3;
        end;
  3..8 :begin
                 for c23:=1 to 3 do
                  begin
                   lis1[c23]:=13+c23;
                   lis2[c23]:=19+c23;
                   lis3[c23]:=26+c23;
                   lisss2[c23]:=16+c23;
                   lisss3[c23]:=33-c23;
                  end;
                 lisss1[1]:=10;
                 lisss1[2]:=12;
                 lisss1[3]:=13;
                 liss1[1]:=9;
                 liss1[2]:=11;
```

```
                for c23:=1 to 2 do liss2[c23]:=22+c23;
                for c23:=1 to 2 do liss3[c23]:=24+c23;
                sorteo4;
                sorteo2;
                sorteo3;
        end;

 9..10  :begin
                for c23:=1 to 3 do
                 begin
                  lis1[c23]:=13+c23;
                  lis3[c23]:=28-c23;
                  lisss1[c23]:=10+c23;
                  lisss3[c23]:=33-c23;
                 end;
                lisss2[1]:=17;
                lisss2[2]:=18;
                lisss2[3]:=20;
                lis2[1]:=19;
                lis2[2]:=21;
                lis2[3]:=22;
                for c23:=1 to 2 do liss1[c23]:=8+c23;
                for c23:=1 to 2 do liss2[c23]:=22+c23;
                for c23:=1 to 2 do liss3[c23]:=27+c23;
                sorteo1;
                sorteo2;
                sorteo3;
        end;
 11     :begin
                for c23:=1 to 3 do
                 begin
                  lis2[c23]:=19+c23;
                  lis3[c23]:=28-c23;
                  lisss2[c23]:=16+c23;
                  lisss3[c23]:=33-c23;
                 end;
                for c23:=1 to 2 do liss1[c23]:=8+c23;
                for c23:=1 to 2 do liss2[c23]:=22+c23;
                for c23:=1 to 2 do liss3[c23]:=27+c23;
                lisss1[1]:=12;
                lisss1[2]:=11;
                lisss1[3]:=14;
                lis1[1]:=13;
                lis1[2]:=15;
                lis1[3]:=16;
                sorteo5;
                sorteo2;
                sorteo6;
        end;
 12..14:begin
                for c23:=1 to 3 do
                 begin
                  lis1[c23]:=13+c23;
                  lis3[c23]:=28-c23;
                  lisss1[c23]:=10+c23;
                  lisss2[c23]:=16+c23;
                  lisss3[c23]:=27+c23;
                 end;
                for c23:=1 to 2 do liss1[c23]:=8+c23;
                for c23:=1 to 2 do liss3[c23]:=30+c23;
                liss2[1]:=22;
                liss2[2]:=24;
                lis2[1]:=20;
```

```
                lis2[2]:=21;
                lis2[3]:=23;
                sorteo7;
                sorteo8;
                sorteo9;
        end;
  15..20:begin
                for c23:=1 to 3 do
                 begin
                  lis1[c23]:=13+c23;
                  lis3[c23]:=28-c23;
                  lisss1[c23]:=10+c23;
                  lisss2[c23]:=16+c23;
                  lisss3[c23]:=27+c23;
                 end;
                for c23:=1 to 2 do liss1[c23]:=8+c23;
                for c23:=1 to 2 do liss3[c23]:=30+c23;
                liss2[1]:=20;
                liss2[2]:=23;
                lis2[1]:=21;
                lis2[2]:=22;
                lis2[3]:=24;
                sorteo10;
                sorteo2;
                sorteo9;
          end;

 end;
 end;
close(datos);
end.
```