1. Desarrollo de la estrategia

Con la idea de desarrollar un algoritmo para el dataset recibido, comienzo analizando la distribución de los grupos familiares, los bonos que se van a pagar, y la distribución de los dias preferidos de las familias.

Grupos familiares

GF	Total familias	Total personas
2	717	1434
3	981	2943
4	1451	5804
5	899	4495
6	494	2964
7	301	2107
8	157	1256
	5000	**21003**

Teniendo una capacidad disponible de 34.000, no pienso que vaya a ser un problema asignar todas las familias. Me enfoco desde el comienzo en asignar las familias para minimizar el costo de los bonos.

Bonos

Costos

GF	Pref 1	Pref 2	Pref 3	Pref 4	Pref 5	Pref 6	Pref 7
2	$50.00	$55.00	$60.00	$65.00	$70.00	$75.00	$80.00
3	$60.00	$65.00	$70.00	$75.00	$80.00	$85.00	$90.00
4	$70.00	$75.00	$80.00	$85.00	$90.00	$95.00	$100.00
5	$80.00	$85.00	$90.00	$95.00	$100.00	$105.00	$110.00
6	$90.00	$95.00	$100.00	$105.00	$110.00	$115.00	$120.00
7	$100.00	$105.00	$110.00	$115.00	$120.00	$125.00	$130.00
8	$110.00	$115.00	$120.00	$125.00	$130.00	$135.00	$140.00

Del costo de los bonos se desprenden dos posibles estrategias:

a. Pagar menos bonos, de mayor valor, asignando primero las familias más chicas.
b. Pagar más bonos, de menor valor, asignando primero las familias más grandes.

Ahorro por asignar los 340 lugares a cada grupo familiar

GF	Total familias	Pref 1	Total	Pref 7	Total
2	170	$50.00	$8,500.00	$80.00	$13,600.00
3	113	$60.00	$6,780.00	$90.00	$10,170.00
4	85	$70.00	$5,950.00	$100.00	$8,500.00
5	68	$80.00	$5,440.00	$110.00	$7,480.00
6	56	$90.00	$5,040.00	$120.00	$6,720.00
7	48	$100.00	$4,800.00	$130.00	$6,240.00
8	42	$110.00	$4,620.00	$140.00	$5,880.00

Se ahorra más asignando los grupos familiares más chicos, que los más numerosos. Es más costoso pagar más bonos, de menor valor, que pagar bonos más caros, en menor cantidad.

Primeras preferencias

Asignando todas las familias a su dia preferido, resulta que hay 84 dias con capacidad suficiente, para un total de 3,273 familias (13,715 personas), con un promedio de 176 lugares libres.

Los dias con capacidad insuficiente son 16, para un total de 1,727 familias (7,288 personas). Sacando el día más requerido (dia 1, con 1,576 personas anotadas), los demás días tienen un promedio de exceso de 22 personas.

El costo de los bonos a pagar va a depender de como se asignen las familias en los días con exceso, principalmente las familias del día 1, que define la mayoría de los bonos a pagar.

Distribución de los grupos familiares en los días sobrepasados

Día	GF 2	GF 3	GF 4	GF 5	GF 6	GF 7	GF 8	Total personas
1	64	59	114	57	35	24	19	1576
3	15	15	30	15	9	8	4	412
4	14	9	27	11	17	5	3	379
5	10	18	18	19	11	5	2	358
10	10	23	21	15	8	6	3	362
11	11	23	23	18	5	6	6	393
12	13	18	27	18	11	11	2	497
17	17	19	28	29	8	3	2	433

19	12	17	25	15	9	4	3	356
25	13	21	12	30	13	3	4	418
26	12	22	30	16	7	2	3	370
31	9	14	28	18	6	5	3	357
32	13	14	21	15	7	7	5	358
33	14	17	30	22	8	2	1	379
45	17	11	31	15	4	4	4	350
59	16	19	25	16	6	3	3	350

Carga aproximada de las familias (grupos familiares sin asignar)

Desde los grupos familiares más grandes

Día	GF 2	GF 3	GF 4	GF 5	GF 6	GF 7	GF 8	Total personas
1	64	59	114	57	31	0	0	344
3	15	11	0	0	0	0	0	340
4	14	3	0	0	0	0	0	342
5	9	0	0	0	0	0	0	340
10	10	0	0	0	0	0	0	342
11	11	10	0	0	0	0	0	341
12	13	18	4	0	0	0	0	341
17	17	19	0	0	0	0	0	342
19	8	0	0	0	0	0	0	340
25	13	17	0	0	0	0	0	341
26	12	2	0	0	0	0	0	340
31	8	0	0	0	0	0	0	341
32	9	0	0	0	0	0	0	340
33	14	3	0	0	0	0	0	342
45	5	0	0	0	0	0	0	340
59	5	0	0	0	0	0	0	340
Costo mínimo de los bonos a pagar:					**$35,660.00**			

Desde los grupos familiares más chicos

Día	GF 2	GF 3	GF 4	GF 5	GF 6	GF 7	GF 8	Total personas
1	0	0	105	57	35	24	19	341
3	0	0	0	0	0	5	4	345
4	0	0	0	0	0	2	3	341
5	0	0	0	0	0	0	2	342
10	0	0	0	0	0	0	2	346
11	0	0	0	0	0	0	6	345
12	0	0	0	0	0	11	2	344
17	0	0	0	1	8	3	2	343
19	0	0	0	0	0	0	2	340
25	0	0	0	0	4	3	4	341
26	0	0	0	0	0	0	3	346
31	0	0	0	0	0	0	3	341
32	0	0	0	0	0	0	2	342
33	0	0	0	0	2	2	1	345
45	0	0	0	0	0	0	1	342
59	0	0	0	0	0	0	1	342
Costo mínimo de los bonos a pagar:					**$27,560.00**			

Consideraciones finales

Como un algoritmo *greedy* no puede probar combinaciones para elegir la mejor solución, la idea es que recorra las familias de manera tal que asigne primero las familias que minimicen el costo total de los bonos a pagar; es decir, que minimicen la cantidad de bonos pagados. Para lograr eso, voy a recorrer las familias desde las de menor tamaño, para asignar la mayor cantidad posible de familias a sus dias preferidos.

Problema. Según el orden que se asignen los días, las segundas preferencias pueden llenar días completables con primeras preferencias.

Solución. Ir asignando las familias por orden de preferencia: primero asignar las preferencias 0, luego las preferencias 1, etc.

2. Implementación del algoritmo

Estrategia. Como la idea es pagar la menor cantidad posible de bonos, conviene recorrer las familias en orden ascendente de tamaño, y tratar de asignarlas en su día preferido. Si al final del recorrido quedan familias sin asignar, volver a recorrerlas y tratar de asignarlas a un dia de preferencia mayor, y así hasta que no queden familias para asignar, o se haya pasado por todos los días preferidos. En este caso, el algoritmo no va a haber encontrado una solución.

Estructuras auxiliares. Como para asignar una familia a un dia preferido, necesito saber si ese día se sobrepasa al agregar la familia, mantener un array de los días disponibles donde el índice representa el día $[1 \rightarrow 100]$, y el valor representa la cantidad de personas asignadas a ese dia.

Pseudocódigo

```
P = 0 // preferencia a asignar
dias[100]

Mientras haya familias para asignar y P sea menor a TOT_PREFERENCIAS {

      Recorrer Familias por tamaño ascendente {

            Si la familia entra en el dia de preferencia P
                  → agregar Familia a la solución
                  → borrar Familia de Familias
      }

      P++
}
```

Implementación

Para poder acceder a las familias por tamaño de grupo familiar, las guardo en un TreeMap con claves : tamaño grupo familiar, y valores: array de familias.

Probando la solución propuesta: no hay solución

Costo total: 29905
Familias sin asignar (1):
ID: 2889; Miembros: 5; Dias preferidos: [17, 32, 4, 52, 26, 5, 10, 18]

Corrección

Es necesario mejorar el recorrido de las familias para priorizar las que tengan más dificultad en asignarse en días de preferencia mayor. Ordenar las familias dando más prioridad a las familias cuya cantidad de personas asignadas en su dia de preferencia siguiente (al asignado en ese ciclo) sea mayor.

Con la introducción de este tipo de ordenamiento, conviene asignar primero las familias cuyos dias preferidos no se sobrepasan, de manera tal que las familias cuyos dias preferidos se sobrepasan puedan ser ordenadas desde su primer asignación.

Pseudocódigo

```
Ordenamiento (Familia f1, Familia f2, pref, dias[]) {
      IGUALES si dias[f1.diaEn(pref+1)] == dias[f2.diaEn(pref+1)]
      F1 es MAYOR si dias[f1.diaEn(pref+1) < dias[f2.diaEn(pref+1)]
      F1 es MENOR si dias[f1.diaEn(pref+1)] > dias[f2.diaEn(pref+1)]
}

dias[100]

Calcular que dias no se sobrepasan con las primeras preferencias.
Asignar las familias en los dias que no se sobrepasan.

P = 0
Mientras haya familias para asignar y P sea menor a TOT_PREFERENCIAS {

      Para cada grupo familiar de menor a mayor {

            Ordenar grupo familiar

            Para cada familia del grupo familiar {

                  Si la familia entra en el dia de preferencia P
                        → agregar Familia a la solución
                        → borrar Familia de Familias
            }
      }

      P++
}
```

Probando la solución propuesta: hay solución

Costo total: 29790

3. Desarrollo e implementación de la mejora

Si bien asignar las familias a sus dias de preferencia por orden ascendente de tamaño garantizó pagar la menor cantidad de bonos posibles, en algunos casos hubiese sido mejor asignar una familia de mayor tamaño, ya sea porque:

1. en el día quedaron lugares libres que no pudieron ser completados, lo que hubiese permitido pagar un bono de menor valor, si se hubiese asignado la familia de mayor tamaño ocupando los lugares vacíos
2. a la familia que tuvo prioridad en la asignación se le hubiese podido asignar un dia de menor preferencia del que terminó siendo asignado a otras familias con menos prioridad en el momento del recorrido (el ordenamiento controlaba que el dia de preferencia siguiente tenga mayor capacidad ocupada, ANTES de empezar las asignaciones de ese grupo familiar, sin tener en cuenta (1) las que fueron asignadas en ese grupo familiar, y (2) el estado de los demás dias preferidos)

El objetivo de la mejora es buscar, para todas las familias [A] a las que no le fue asignado su dia preferido, una familia [B] (asignada en algun dia de preferencia menor al asignado a A) que si se asignase a un dia de preferencia mayor, reciba un bono menor que el recibido por A.

Estrategia. Recorrer las familias [A] a las que no les fue asignado su dia preferido. Para cada dia de preferencia menor al asignado, recorrer las familias [B] asignadas ese dia (tomadas desde el mínimo grupo familiar que le permita a A ser asignada ese día), y buscar en los dias de preferencia mayores al que tiene asignado B, un día al que B pueda ser asignada tal que la suma de los nuevos bonos que reciben A y B, sea menor que la que recibian antes del intercambio. Encontrar la familia B que garantice el mayor ahorro.

Implementación

Necesito estructurar la solución para poder acceder a las familias en orden de preferencia asignada, y por grupos familiares. Utilizo un TreeMap con clave : preferencia asignada, y valor : un TreeMap con clave : grupo familiar, y valor : HashSet de familias.

Pseudocódigo

```
Por cada dia asignado {

      Por cada familia(A) asignada con preferencia(Pa) mayor a 0 {

            Costoa = bono que recibe A
            Ahorro = 0 // inicializado en 0
            FamiliaIntercambiable = null // familia que da el mayor ahorro

            Por cada dia(Da) con preferencia(Pn) desde Pa-1 hasta 0 {

                  T = menor grupo familiar que permita a A entrar en Da

                  Para cada familia(B) asignada en Da desde tamaño T {

                        Costob = bono que recibe B
                        CostoTotal = Costoa + Costob
                        Pb = preferencia asignada a B

                        Por cada dia(Db) con preferencia(Pm) desde Pb+1 hasta 8 {

                              Si B entra en Db {

                                    NuevoCostoa = bono que recibe A en Da
                                    NuevoCostob = bono que recibe B en Db
                                    NuevoAhorro = costoTotal–NuevoCostoa-NuevoCostob

                                    Si NuevoAhorro > Ahorro {
                                          // se encontro un intercambio mejor
                                          Ahorro = NuevoAhorro
                                          FamiliaIntercambiable = B
                                    }
                              }
                        }
                  }
            }

            Si se encontro una FamiliaIntercambiable {
                  Intercambiar
            }
      }
}
```

Probando la solución propuesta

Iteraciones: 7989; Intercambios: 29

Costo total: 29395

Objetos implementados para simplificar las operaciones del algoritmo