Guía 4

Rodrigo Estay Muñoz

IMPORTANTE: Ya que debe ser una flojera escribir todo este código para revisarlo, aquí esta un link con el código en Github:

https://github.com/RodrigoEstay/Programacion-I/blob/master/guia5/problema1.c

problema 1:

```
 8  // Definimos las estructuras como se nos pidio:
 9  typedef struct Fanatico{
10      char nombre[100];
11      unsigned char edad;
12      double agresividad;
13  }Fanatico;
14
15  typedef struct Jugador{
16      char nombre[100];
17      unsigned char edad;
18      char posicion[20];
19      double regate;
20      double defensa;
21      double velocidad;
22      double dureza;
23      double resistencia;
24      double reflejos;
25  }Jugador;
26
27  typedef struct Staff{
28      char nombre[100];
29      unsigned char edad;
30      char rol[20];
31      unsigned char experiencia;
32  }Staff;
33
```

```
33
34  typedef struct Equipo{
35      char nombre[100];
36      int anyoFundacion;
37      struct Staff *staff;
38      int numStaff[6];
39      struct Jugador *jugadores;
40      int numJugadores[4];
41      struct Fanatico *fanaticos;
42      int numFanaticos;
43  }Equipo;
44
```

Problema 2:

```
119 // En esta funcion solo se generan los valores aleatoriamente y se lo asignamos a los fanaticos.
120 // La funcion generarNombre devuelve un nombre y apellido aleatorio.
121 Fanatico* crearFanaticos(int numFanaticos){
122     Fanatico *fanaticos;
123     fanaticos=(Fanatico*)malloc(numFanaticos*sizeof(Fanatico));
124     int i;
125     char nombre[100];
126     for(i=0;i<numFanaticos;++i){
127         generarNombre(nombre, rand()%30, rand()%30);
128         strcpy((fanaticos+i)->nombre, nombre);
129         (fanaticos+i)->edad=rand()%100+1; // Edad desde 1-100.
130         (fanaticos+i)->agresividad=(double)rand()/100;
131     }
132     return fanaticos;
133 }
134
135 // Se libera su memoria.
136 void liberarFanaticos(Fanatico* fans){
137     free(fans);
138 }
```

```
// Creamos los miembros del staff con la cantidad de roles especificada, y luego le asignamos valores
// aleatrios de una manera parecida que a los fanaticos.
Staff* crearStaff(int numDT, int numPrepArq, int numPrepFis, int numAsis, int numFisio, int numDoc){
    int i, total=numDT+numPrepFis+numPrepArq+numAsis+numFisio+numDoc;
    Staff *staff=(Staff*)malloc(total*sizeof(Staff));
    char nombre[100];
    // En este ciclo se le va asignando el rol a cada miembro del staff creado, observamos
    // que por cada vez que se crea el staff se ejecuta un if y luego se disminuye su numero,
    // de esta manera cuando el numero llegue a 0 se dejaran de crear staff con ese rol.
    for(i=0;i<total;++i){
        if(numDT){
            strcpy((staff+i)->rol, "DT");
            --numDT;
        }
        else if(numPrepArq){
            strcpy((staff+i)->rol, "Preparador Arqueros");
            --numPrepArq;
        }
        else if(numPrepFis){
            strcpy((staff+i)->rol, "Preparador Fisico");
            --numPrepFis;
        }
        else if(numAsis){
            strcpy((staff+i)->rol, "Asistente");
            --numAsis;
        }
        else if(numFisio){
            strcpy((staff+i)->rol, "Fisioterapeuta");
            --numFisio;
        }
        else if(numDoc){
```

```
            else if(numDoc){
                strcpy((staff+i)->rol, "Doctor");
                --numDoc;
            }
            generarNombre(nombre, rand()%30, rand()%30);
            strcpy((staff+i)->nombre, nombre);
            (staff+i)->edad=rand()%75+20; // Edad desde 20-74.
            (staff+i)->experiencia=rand()%256;
        }
        return staff;
}

// Liberamos la memoria del staff.
void liberarStaff(Staff* staff){
    free(staff);
}
```

```
187 // Se crean numeros aleatorios para los distintos valores de los jugadores, y si le generan
188 // nombres, la cantidad de jugadores creados para cada posicion es igual a como se hizo
189 // en el staff. Cabe notar que los valores generados aleatoriamente para los de coma flotante
190 // de precision doble son divididos por numeros muy altos, es para evitar que estos valores
191 // sean muy grandes, ya que asi despues no se pueden imprimir ordenadamente.
192 Jugador* crearJugadores(int numArq, int numDef, int numCent, int numDel){
193     int i, total=numArq+numDef+numCent+numDel;
194     Jugador *jugadores=(Jugador*)malloc(total*sizeof(Jugador));
195     char nombre[100];
196     for(i=0;i<total;++i){
197         if(numArq){
198             strcpy((jugadores+i)->posicion, "Arquero");
199             --numArq;
200         }
201         else if(numDef){
202             strcpy((jugadores+i)->posicion, "Defensa");
203             --numDef;
204         }
205         else if(numCent){
206             strcpy((jugadores+i)->posicion, "Medio");
207             --numCent;
208         }
209         else if(numDel){
210             strcpy((jugadores+i)->posicion, "Delantero");
211             --numDel;
212         }
213         generarNombre(nombre, rand()%30, rand()%30);
214         strcpy((jugadores+i)->nombre, nombre);
215         (jugadores+i)->edad=rand()%13+18; // Edad desde 18-30.
216         (jugadores+i)->regate=(double)rand()/10000000;
217         (jugadores+i)->defensa=(double)rand()/1000000;
218         (jugadores+i)->reflejos=(double)rand()/1000000;
```

```
217             (jugadores+i)->defensa=(double)rand()/1000000;
218             (jugadores+i)->reflejos=(double)rand()/1000000;
219             (jugadores+i)->velocidad=(double)rand()/1000000;
220             (jugadores+i)->dureza=(double)rand()/1000000;
221             (jugadores+i)->resistencia=(double)rand()/1000000;
222         }
223         return jugadores;
224 }
225
226 // Se libera la memoria de los jugadores.
227 void liberarJugadores(Jugador* jugadores){
228     free(jugadores);
229 }
230
```

```
231 // Se crean los equipos con nombres aleatorios.
232 Equipo* crearEquipos(int numEquipos){
233     Equipo *equipos=(Equipo*)malloc(numEquipos*sizeof(Equipo));
234     int i, j;
235     for(i=0;i<numEquipos;++i){
236         strcpy((equipos+i)->nombre, rNombreEquipos[rand()%20]);
237         // Para los primeros 20 equipos nos aseguramos de que sus nombres sean distintos.
238         // Solo nos preocupamos de los primeros 20 porque nuestra variable global con
239         // nombres para los equipos solo tiene 20 nombres posibles, asi que pasados
240         // los 20 nombres generados, se pueden empezar a repetir.
241         if(i<20){
242             for(j=0;j<i;++j){
243                 if(!strcmp((equipos+j)->nombre, (equipos+i)->nombre)){
244                     strcpy((equipos+i)->nombre, rNombreEquipos[rand()%19]);
245                     j=-1;
246                 }
247             }
248         }
249         // Año de fundacion desde 1800-2018.
250         (equipos+i)->anyoFundacion=1800+rand()%219;
251     }
252     return equipos;
253 }
254
```

```
255 // Para cada equipo liberamos la memoria de todos sus vectores, y luego la de los equipos.
256 void liberarEquipos(Equipo* equipos, int numEquipos){
257     int i;
258     for(i=0;i<numEquipos;++i){
259         liberarFanaticos((equipos+i)->fanaticos);
260         liberarJugadores((equipos+i)->jugadores);
261         liberarStaff((equipos+i)->staff);
262     }
263     free(equipos);
264 }
```

Nombres y función adicional:

```
// Estas variables globales son para definir nombres aleatorios de tanto personas como equipos.

char rNombres[30][15]={"Pedro", "Antonio", "Angela", "Maria", "Jose", "Miguel", "Paula",
                "Catalina", "Mario", "Andres", "Valentina", "Josefa", "Alexis", "Hugo",
                "Paola", "Angelica", "Rodrigo", "Ignacio", "Fernanda", "Camila","Cristobal",
                "Martin", "Ignacia", "Valeria", "Carlos", "Silvana", "Sofia", "Benjamin",
                "Vicente", "Matias"};
char rApellidos[30][15]={"Gonzales", "Munoz", "Rojas", "Dias", "Perez", "Soto", "Silva",
                "Contreras", "Lopez", "Rodriguez","Morales", "Martinez", "Fuentes", "Estay",
                "Araya", "Sepulveda", "Espinoza", "Torres", "Castillo","Reyes", "Freire",
                "Ruiz", "Pino", "Toro", "Correa", "Medina", "Pinto", "Venegas", "Acevedo", "Salas"};
char rNombreEquipos[20][23]={"Los Leones", "Los Dragones", "Forestal sur", "Bio-Bio FC", "Los Lagartos",
                "Los Invencibles", "Los Vencedores", "Maipu FC", "Los del Pueblo", "Los Renguinos",
                "CD Alerce", "Los Imparables","Saturno", "Azukol", "Deportes Rengo", "Master",
                "Olimpicos", "Los Perfumados", "Condores Negros","Flamencos Endemoniados"};
```

```
// Esta funcion recibe un string y dos valores, y modifica este string para que sea
// un nombre aleatorio, donde el nombre es generado a partir del primer valor, y el
// apellido es generado a partir del segundo valor.
char* generarNombre(char *nombre, int seedN, int seedA){
    char space[2]=" ";
    strcpy(nombre, rNombres[seedN]);
    strcat(nombre, space);
    strcat(nombre, rApellidos[seedA]);
    return nombre;
}
```

Problema 3:

```
277 // Funcion que ordena segun los criterios especificados
278 void ordenarDB(Equipo* equipo, int numEquipos){
279     int i, j, k, dif, dif2, pos, totalJ, totalS;
280     // Valores temporales para cada estructura para ordenarlos como se especifico.
281     Equipo tempE;
282     Jugador tempJ;
283     Fanatico tempF;
284     Staff tempS;
285     // Este ciclo ordena los nombres de los equipos en orden alfabetico ascendente.
286     for(i=0;i<numEquipos-1;++i){
287         for(j=i+1;j<numEquipos;++j){
288             dif=strcmp((equipo+i)->nombre, (equipo+j)->nombre);
289             if(dif>0){
290                 tempE=equipo[i];
291                 equipo[i]=equipo[j];
292                 equipo[j]=tempE;
293             }
294         }
295     }
296     // Una vez ordenados los equipos comenzamos a ordenar los distintos vectores que posee.
297     for(i=0;i<numEquipos;++i){
298         // Calculamos numero total de jugadores.
299         totalJ=(equipo+i)->numJugadores[0]+(equipo+i)->numJugadores[1]+(equipo+i)->numJugadores[2]
300                 +(equipo+i)->numJugadores[3];
301         // En este ciclo se ordenan los jugadores, primero por posicion en orden alfabetico.
302         for(j=0;j<totalJ-1;++j){
303             for(k=j+1;k<totalJ;++k){
304                 dif=strcmp(((equipo+i)->jugadores +j)->posicion, ((equipo+i)->jugadores +k)->posicion);
305                 if(dif>0){
306                     tempJ=((equipo+i)->jugadores)[j];
307                     ((equipo+i)->jugadores)[j]=((equipo+i)->jugadores)[k];
308                     ((equipo+i)->jugadores)[k]=tempJ;
```

```
307                     ((equipo+i)->jugadores)[j]=((equipo+i)->jugadores)[k];
308                     ((equipo+i)->jugadores)[k]=tempJ;
309                 }
310                 else if(dif==0){ // Si poseen la misma posicion, se ordena por nombre.
311                     dif2=strcmp(((equipo+i)->jugadores +j)->nombre, ((equipo+i)->jugadores +k)->nombre);
312                     if(dif2>0){
313                         tempJ=((equipo+i)->jugadores)[j];
314                         ((equipo+i)->jugadores)[j]=((equipo+i)->jugadores)[k];
315                         ((equipo+i)->jugadores)[k]=tempJ;
316                     }
317                 }
318             }
319         }
320         // Este ciclo ordena los fanaticos por agresividad (ascendentemente).
321         for(j=0;j<(equipo+i)->numFanaticos -1;++j){
322             for(k=j+1;k<(equipo+i)->numFanaticos;++k){
323                 if(((equipo+i)->fanaticos +j)->agresividad > ((equipo+i)->fanaticos +k)->agresividad){
324                     tempF=((equipo+i)->fanaticos)[j];
325                     ((equipo+i)->fanaticos)[j]=((equipo+i)->fanaticos[k]);
326                     ((equipo+i)->fanaticos)[k]=tempF;
327                 }
328                 // Si poseen el mismo valor de agresividad, se ordenan por nombre.
329                 else if(((equipo+i)->fanaticos +j)->agresividad == ((equipo+i)->fanaticos +k)->agresividad){
330                     dif=strcmp(((equipo+i)->fanaticos +j)->nombre, ((equipo+i)->fanaticos +j)->nombre);
331                     if(dif>0){
332                         tempF=((equipo+i)->fanaticos)[j];
333                         ((equipo+i)->fanaticos)[j]=((equipo+i)->fanaticos)[k];
334                         ((equipo+i)->fanaticos)[k]=tempF;
335                     }
336                 }
337             }
338         }
```

```
            }
        }
        // Calculamos el numero total de staff.
        totalS=(equipo+i)->numStaff[0]+(equipo+i)->numStaff[1]+(equipo+i)->numStaff[2]+(equipo+i)->numStaff[3]
            +(equipo+i)->numStaff[4]+(equipo+i)->numStaff[5];
        // Este ciclo ordena el staff por su rol (alfabeticamente).
        for(j=0;j<totalS-1;++j){
            for(k=j+1;k<totalS;++k){
                dif=strcmp(((equipo+i)->staff +j)->rol, ((equipo+i)->staff +k)->rol);
                if(dif>0){
                    tempS=((equipo+i)->staff)[j];
                    ((equipo+i)->staff)[j]=((equipo+i)->staff)[k];
                    ((equipo+i)->staff)[k]=tempS;
                }
                else if(dif==0){ // Si poseen el mismo rol, se ordena por nombre.
                    dif2=strcmp(((equipo+i)->staff +j)->nombre, ((equipo+i)->staff +k)->nombre);
                    if(dif2>0){
                        tempS=((equipo+i)->staff)[j];
                        ((equipo+i)->staff)[j]=((equipo+i)->staff)[k];
                        ((equipo+i)->staff)[k]=tempS;
                    }
                }
            }
        }
    }
}
```

Problema 4:

```
364 // Esta funcion simplemente imprime los datos de la manera mas ordenada posible.
365 void imprimiDB(Equipo* equipo, int numEquipos){
366     int i, j, totalS, totalJ;
367     for(i=0;i<numEquipos;++i){
368         printf("Equipo: %s\nFundado en: %d\n\n", (equipo+i)->nombre, (equipo+i)->anyoFundacion);
369         // Calculamos total de juadores.
370         totalJ=(equipo+i)->numJugadores[0]+(equipo+i)->numJugadores[1]+(equipo+i)->numJugadores[2]
371                 +(equipo+i)->numJugadores[3];
372         printf("Jugadores:\n");
373         // Los espacios en blanco son considerando los espacios que tomaran los string.
374         printf("        Nombre     Edad Posicion Regate\tDefensa\tReflej\tVelo\tDureza\tResisten\n");
375         for(j=0;j<totalJ;++j){
376             // Hacemos que siempre ocupen una cantidad de espacios en blancos para mantener el orden.
377             printf("%19s %2u %9s %5.2lf\t%5.2lf\t%5.2lf\t%5.2lf\t%5.2lf\t%5.2lf\n",
378                 ((equipo+i)->jugadores +j)->nombre, ((equipo+i)->jugadores +j)->edad,
379                 ((equipo+i)->jugadores +j)->posicion, ((equipo+i)->jugadores +j)->regate,
380                 ((equipo+i)->jugadores +j)->defensa, ((equipo+i)->jugadores +j)->reflejos,
381                 ((equipo+i)->jugadores +j)->velocidad, ((equipo+i)->jugadores +j)->dureza,
382                 ((equipo+i)->jugadores +j)->resistencia);
383         }
384         // Calculamos el numero total de staff y se imprimen de manera parecida que los jugadores.
385         totalS=(equipo+i)->numStaff[0]+(equipo+i)->numStaff[1]+(equipo+i)->numStaff[2]+(equipo+i)->numStaff[3]
386             +(equipo+i)->numStaff[4]+(equipo+i)->numStaff[5];
387         printf("\nStaff:\n");
388         printf("        Nombre     Edad         Rol         Experiencia\n");
389         for(j=0;j<totalS;++j){
390             printf("%19s %3u %19s %3u\n", ((equipo+i)->staff +j)->nombre, ((equipo+i)->staff +j)->edad,
391                 ((equipo+i)->staff +j)->rol, ((equipo+i)->staff +j)->experiencia);
392         }
```

```
391                 ((equipo+i)->staff +j)->rol, ((equipo+i)->staff +j)->experiencia);
392         }
393         printf("\nFanaticos:\n");
394         printf("        Nombre     Edad Agresividad\n");
395         for(j=0;j<(equipo+i)->numFanaticos;++j){
396             printf("%19s %3u %5.3lf\n", ((equipo+i)->fanaticos +j)->nombre, ((equipo+i)->fanaticos +j)->edad,
397                 ((equipo+i)->fanaticos +j)->agresividad);
398         }
399         // Imprimimos dos saltos de linea a no ser que se trate del ultimo equipo impreso.
400         if(i<numEquipos-1){
401             printf("\n\n");
402         }
403     }
404 }
```

Problema 5:

```
73  int main(){
74      system("clear");
75      int numEquipos, maxFanaticos, minFanaticos, numFanaticos, i;
76      // Decalaramos srand esta unica vez para generar todos los casos aleatorios.
77      srand(time(NULL));
78      printf("Ingrese el numero de equipos y el numero minimo y maximo de fanaticos.\n");
79      scanf("%d%d%d", &numEquipos, &minFanaticos, &maxFanaticos);
80      Equipo* equipo=crearEquipos(numEquipos);
81      // Para cada equipo le asignamos un numero aleatorio de fanaticos con los limites ingresados,
82      // creamos el numero de staff y jugadores indicados, luego al equipo le asignamos el numero
83      // de jugadores  y de staff de cada tipo que va a tener y de fans, luego le asignamos los punteros
84      // generados al principio.
85      for(i=0;i<numEquipos;++i){
86          numFanaticos=rand()%(maxFanaticos-minFanaticos+1)+minFanaticos;
87          Fanatico* fans=crearFanaticos(numFanaticos);
88          Jugador* jug=crearJugadores(3, 6, 7, 6);
89          Staff* staff=crearStaff(1, 1, 1, 2, 2, 1);
90          equipo[i].staff=staff;
91          equipo[i].numStaff[0]=1;
92          equipo[i].numStaff[1]=1;
93          equipo[i].numStaff[2]=1;
94          equipo[i].numStaff[3]=2;
95          equipo[i].numStaff[4]=2;
96          equipo[i].numStaff[5]=1;
97          equipo[i].jugadores=jug;
98          equipo[i].numJugadores[0]=3;
99          equipo[i].numJugadores[1]=6;
100         equipo[i].numJugadores[2]=7;
101         equipo[i].numJugadores[3]=6;
102         equipo[i].fanaticos=fans;
103         equipo[i].numFanaticos=numFanaticos;
104     }
```

```
103         equipo[i].numFanaticos=numFanaticos;
104     }
105     // LLamamos las funciones.
106     system("clear");
107     imprimiDB(equipo, numEquipos);
108     printf("\nPresione Enter para ordenar la lista\n");
109     getchar();
110     getchar();
111     system("clear");
112     ordenarDB(equipo, numEquipos);
113     imprimiDB(equipo, numEquipos);
114     // Liberamos la memoria.
115     liberarEquipos(equipo, numEquipos);
116     return 0;
117 }
```