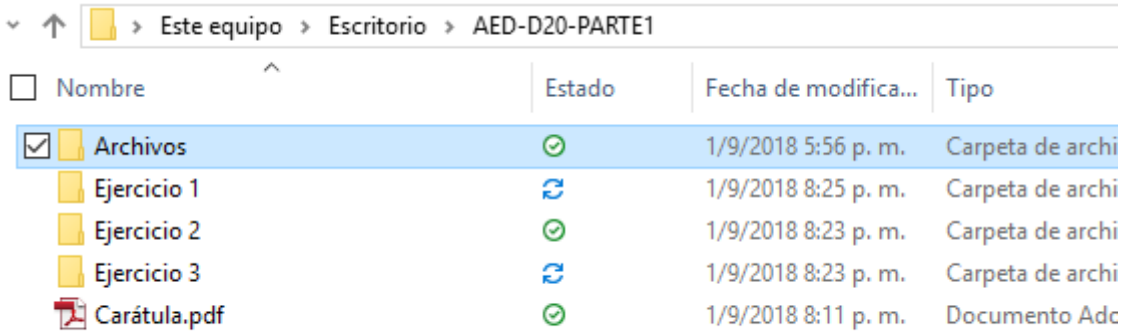
PRECONDICIONES

Para que los ejercicios puedan interactuar entre sí y modificar archivos en forma más prolija, decidimos guardar los ".dat" en una carpeta llamada "Archivos", que debe haber sido creada antes de correr los programas para garantizar su correcto funcionamiento.

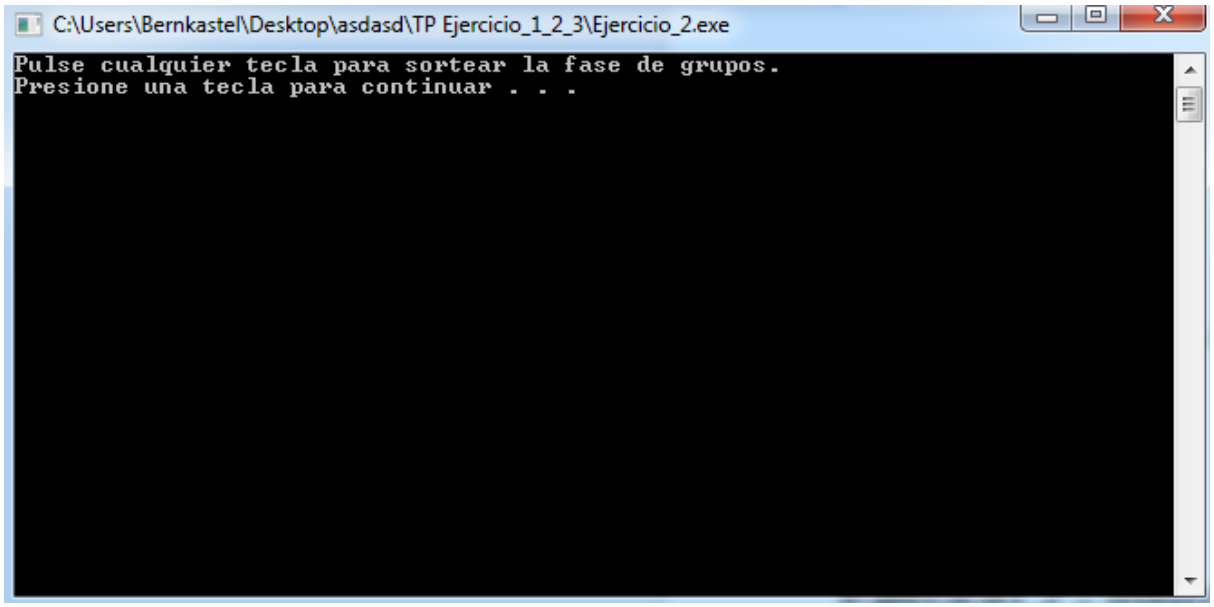


Aclaraciones:

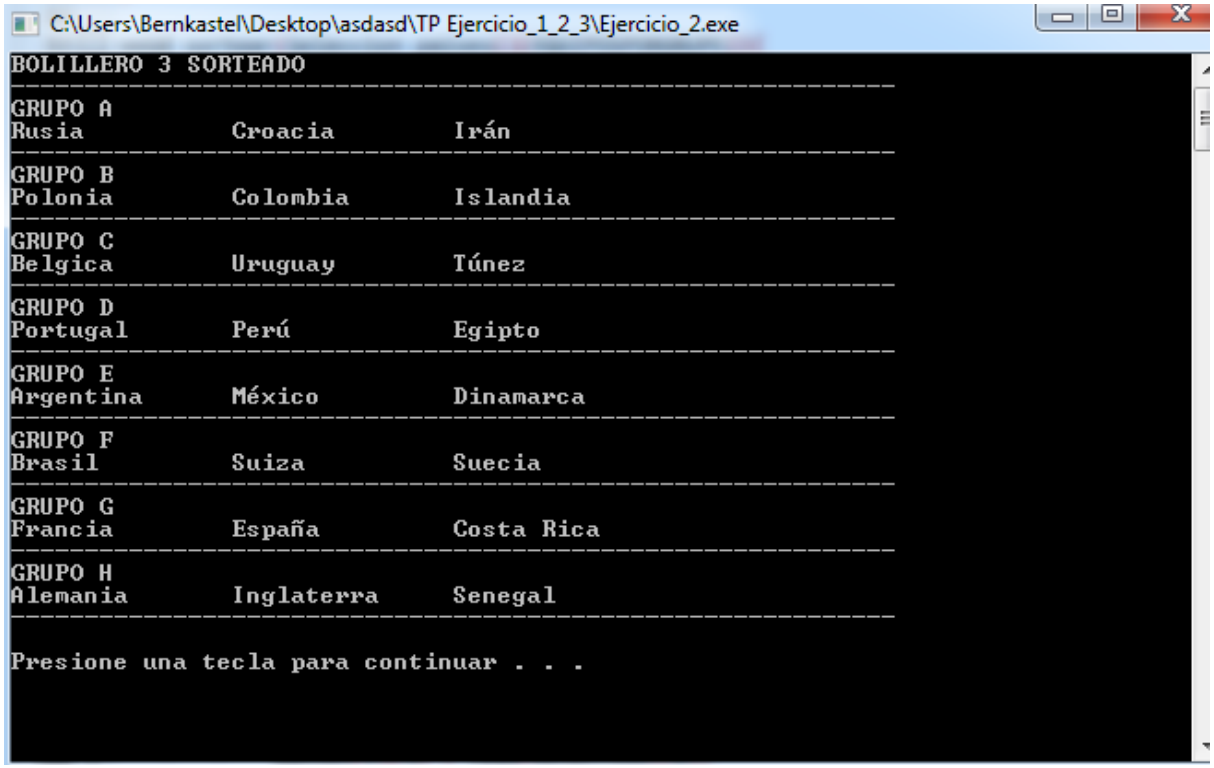
1. No es necesario que exista ningún archivo para que funcione el ejercicio 1.
2. Para que funcione el ejercicio 2, es necesario que existan los archivos generados por el ejercicio 1.
3. Para que funcione el ejercicio 3, es necesario que existan los archivos generados por el ejercicio 2.

EJERCICIO 2 - INSTRUCTIVO

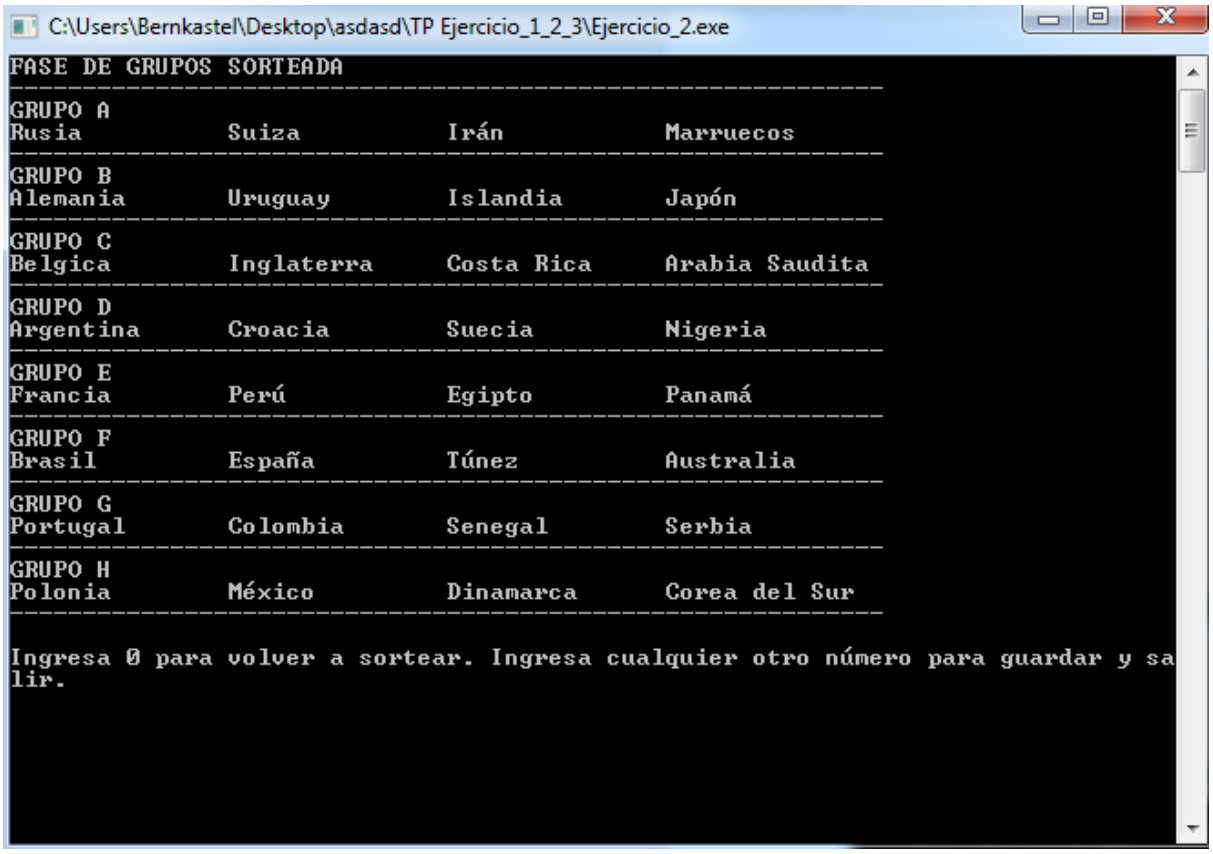
EJECUCIÓN



El programa, en su ejecución solicitará que se presione una tecla.



Con cada accionar se sortearán aleatoriamente los equipos pertenecientes a los bolilleros 1, 2 y 3 hasta conformar los diferentes grupos, siendo Rusia la única selección inamovible.



Al finalizar la conformación de grupos el programa consultará si se desea realizar el sorteo nuevamente en caso de ser necesario.

FUNCIONAMIENTO DEL MAIN (EXPLICACIÓN EN EL RESPECTIVO ORDEN)

```
128 int main() {
129     setlocale(LC_ALL, "");
130     srand(time(NULL));
131     Seleccion paises[GRUPOS][EQUIPOSPORGRUPO];
132 
133     FILE* f = fopen("Equipos/mundial.dat","r+b");
134     leerPaises(f, paises);
135     fclose(f);
136 
137     cout<<"Pulse cualquier tecla para sortear la fase de grupos."<<endl;
138     system("pause");
139 
140     int menu=0;
141     while(menu==0){
142         system("CLS");
143         sortear(paises);
144         cout <<"FASE DE GRUPOS SORTEADA"<<endl;
145         mostrar(paises, EQUIPOSPORGRUPO);
146         cout <<"Ingresa 0 para volver a sortear. Ingresa cualquier otro número para guardar y salir."<< endl;
147         cin>> menu;
148     }
149     guardarGrupos(paises);
150 
151     return 0;
152 }
```

129 y 131: Ídem al ejercicio anterior.

130: Función que permite obtener números aleatorios. Requiere de declarar previamente las librerías “<stdlib.h>” y “<time.h>”.

```
1   #include <iostream>
2   #include <wchar.h>
3   #include <locale.h>
4   #include <stdio.h>
5   #include <stdlib.h>
6   #include <time.h>
7   #include <iomanip>
```

133: Se abre el archivo generado en el ejercicio 1 para leer su contenido.

134: Se invoca la función “leerPaises”, encargada de incorporar los datos de tipo “Seleccion” (que se van recolectando desde el archivo abierto) a la matriz “paises”, utilizando los templates dados por la cátedra.

```
101 void leerPaises(FILE* f, Seleccion paises[][EQUIPOSPORGRUPO]){
102     for(int i=0; i<GRUPOS; i++){
103         for(int j=0; j<EQUIPOSPORGRUPO; j++){
104             paises[i][j] = read<Seleccion>(f);
105         }
106     }
107 }
```

135: Se cierra el archivo

137 y 138: Se imprime por pantalla el menú con la opción de sortear fase de grupos, y se llama a la función system(“pause”) que hace que el usuario tenga que pulsar un botón para continuar con el algoritmo.

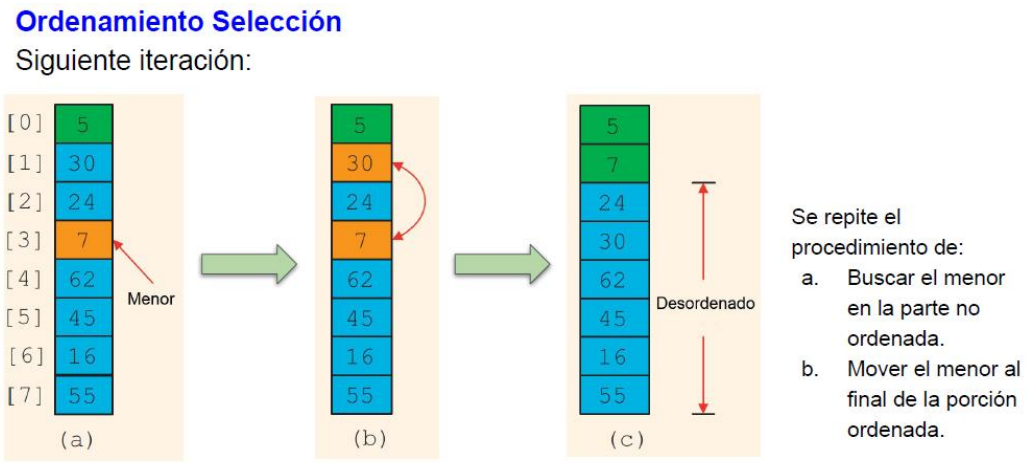
140 y 141: Se declara la variable “menu”, cuyo valor va a ser 0 cuando el usuario quiera sortear, todas las veces que considere necesarias.

142: La función system(“CLS”) se encargará de limpiar la pantalla para una visualización más prolija de lo que se va imprimiendo por pantalla.

143: Se invoca la función “sortear”, que recibe la matriz “paises” y se encarga de reordenarla a medida que se van sorteando los grupos.

```
79  void sortear(Seleccion paises[][EQUIPOSPORGRUPO]){
80      Seleccion temp;
81      int equipoelegido = 0;
82      for (int bol=0; bol<EQUIPOSPORGRUPO; bol++){
83          do {
84              for (int grupo=0; grupo<GRUPOS; grupo++){
85                  if (bol != 0 || grupo != 0){
86                      equipoelegido = rand()%(GRUPOS-grupo) + grupo;
87                      temp = paises[grupo][bol];
88                      paises[grupo][bol] = paises[equipoelegido][bol];
89                      paises[equipoelegido][bol] = temp;
90                  }
91              }
92          } while( verificarEquipos(paises, bol) < 0 );
93          cout <<"BOLILLERO "<< bol+1 << " SORTEADO"<<endl;
94          mostrar( paises, bol+1 );
95          system("pause");
96          system("CLS");
97      }
98      return;
99  }
```

El método de sorteo elegido es similar al ordenamiento por selección (salvo por las líneas 83, 85, 86, 90, 92).



86: En vez de elegirse un índice según cierto criterio (mayor, menor, etc...), se elige a través de un número al azar, en un rango ("GRUPOS-grupo") que va decreciendo de 8 equipos a sortear hasta 1. La suma del final ("+ grupo") se debe a que el equipo elegido no puede ubicarse en las posiciones que ya fueron sorteadas.

85 y 90: Se incorpora para que Rusia (ubicado en la posición [0] [0] de "paises") no sea incluido dentro del sorteo.

83 y 92: Una vez sorteado el bolillero, se deben verificar los criterios de grupo preestablecidos. Para eso, se utiliza la función verificarEquipos, que recibe la matriz "países" y la cantidad de bolilleros sorteados.

```
48  int confederacionCmp(Seleccion e1, string s2)
49  {
50      string s1 = e1.confederacion;
51      return s1>s2?1:s1<s2?-1:0;
52
53  }
54
55  int verificarEquipos(Seleccion paises[][EQUIPOSPORGRUPO], int bolillero){
56
57      int uefa=0, conmebol=0, concacaf=0, afc=0, caf=0, ofc=0;
58
59      for (int grupo=0; grupo<GRUPOS; grupo++){
60          uefa=0, conmebol=0, concacaf=0, afc=0, caf=0, ofc=0;
61
62          for(int i=0; i<=bolillero; i++){
63              if(confederacionCmp(paises[grupo][i], "UEFA") == 0){ uefa++; }
64              if(confederacionCmp(paises[grupo][i], "Conmebol") == 0){ conmebol++; }
65              if(confederacionCmp(paises[grupo][i], "Concacaf") == 0){ concacaf++; }
66              if(confederacionCmp(paises[grupo][i], "AFC") == 0){ afc++; }
67              if(confederacionCmp(paises[grupo][i], "CAF") == 0){ caf++; }
68              if(confederacionCmp(paises[grupo][i], "OFC") == 0){ ofc++; }
69          }
70
71          if( uefa>2 || conmebol>1 || concacaf>1 || afc>1 || caf>1 || ofc>1){
72              return -1;
73          }
74      }
75      return 1;
76  }
```

57: Se declara una variable por cada confederación perteneciente a la FIFA. Éstas se encargan de acumular la cantidad de integrantes de esa confederación en cada grupo.

59, 60 y 69: Se itera en cada grupo, reiniciando las variables declaradas al comenzar.

62 a 69: Se itera entre los bolilleros ya sorteados (incluyendo el último a verificar), para contar la cantidad de integrantes de cada confederación. Se hace llamando a la función “confederacionCmp”, que recibe la “Seleccion” y el “string” que correspondería a la sección “confederacion” del struct. Devuelve “0” en caso de que ambos sean iguales.

71 a 73: La función finaliza y devuelve un valor negativo en caso de que no se cumplan los criterios de grupo (más de 2 integrantes UEFA, o más de 1 de otra confederación). Esto hace que se vuelva a realizar el sorteo de ese mismo bolillero (desde la línea 83).

75: La función devuelve un valor positivo antes de finalizar, ya que se iteraron todos los grupos y no se detectó ningún incumplimiento de los criterios de grupo.

Retornando a “sortear”:

93 a 96: Se muestra por consola que el bolillero fue sorteado exitosamente. La función mostrar está programada para que muestre sólo las columnas de la matriz que ya fueron sorteadas.

```
33  void mostrar(Seleccion paises[][EQUIPOSPORGRUPO], int bolilleros){
34      char grupo = 65;
35      cout<<"----------------------------------------------------------------"<<endl;
36      for(int i=0; i<GRUPOS; i++){
37          cout<< "GRUPO "<< grupo <<endl;
38          for(int j=0; j<bolilleros; j++){
39              cout<< left << setw(15) << paises[i][j].nombreDeEquipo;
40          }
41          cout<<endl;
42          cout<<"----------------------------------------------------------------"<<endl;
43          grupo++;
44      }
45      cout <<endl;
46  }
```

34 y 43: La variable “grupo” de tipo “char” se inicia en la letra ‘A’, y se va incrementando en 1 para formar todas las letras de todos los grupos (B, C, D, E F, G, H).

38 y 44: El “for” itera solamente la cantidad de “bolilleros” que se requieran (en lugar de hacerlo entre todos los “EQUIPOSPORGRUPO”).

39: Ídem al ejercicio anterior.

Retornando al “main”:

144 y 145: Se muestra por consola la totalidad de la matriz “paises”, con el sorteo realizado exitosamente.

146 y 147: Se pregunta al usuario si desea guardar el sorteo o volver a realizarlo.

149: Antes de finalizar, se guarda la matriz “paises” invocando la función “guardarGrupos”

```
109 void guardarGrupos(Seleccion paises[][EQUIPOSPORGRUPO]){
110     FILE* f;
111     string archivo="";
112     string direccion="../Archivos/Grupo ";
113     char grupo='A';
114     string formato=".dat";
115     for(int i=0; i<GRUPOS; i++){
116         archivo=direccion+grupo+formato;
117         const char* c= archivo.c_str();
118         f = fopen(c,"w+b");
119         for(int j=0; j<EQUIPOSPORGRUPO; j++){
120             write<Seleccion>(f, paises[i][j]);
121         }
122         fclose(f);
123         grupo++;
124     }
125     cout<<"Grupos guardados con éxito."<<endl;
126 }
```

111 a 114 y 116 a 117: La variable “archivo” va a contener en formato de string la ruta y nombre del archivo a escribirse (cuyo nombre varía según el grupo). Sin embargo, la función fopen solamente acepta variables tipo const char*. Entonces, se le asigna a la variable “c” el contenido de “archivo” en forma correcta.

118 a 122: Se abre el archivo, se itera entre los distintos equipos del grupo y se escriben utilizando los templates dados por la cátedra. Luego, se cierra el archivo.

123: Se incrementa la variable “grupo” para pasar a la siguiente letra de grupo como ocurre en “mostrar”.