Parcial Algoritmos y Programación II - 7541 Curso Calvo (1º instancia)
1 de junio de 2023

Padrón: **Apellido y Nombre:**

1) Indicar la salida por pantalla y escribir las sentencias necesarias para liberar correctamente la memoria.

```
int main(){
 int *A, *C, *F;
 char *D, *E;
 char **B;
 char G;
 int H;

 H = 66 + ULTIMO_DIGITO_PADRON;
 G = 'A';
 A = new int;
 (*A) = H;
 C = &H;
 F = A;
 H++;
 cout << H << (*A) << (*C) << (*F) << endl;

 D = new char[4];
 E = D + 1;
 B = &D;
 D[0] = G;
 D[1] = 'C';
 cout << (*D) << (*E) << (**B) << endl;

 E[1] = *(D + 1);
 E[2] = G;
 E = &G;
 G = 'B';
 cout << (*B)[2] << (*E) << *(D + 2) << endl;

 E = D;
 D = (char*) F;
 (*A)--;
 cout << (**B) << (*A) << (*D) << endl;

 // liberar la memoria

 return 0;
}
```

2. Diseñar la especificación e implementar el **TDA Cocina** con las siguientes operaciones:

- Crearlo a partir de una cantidad de cocineros
- Iniciar el preparado de un plato (tiene que haber un cocinero libre), devuelve el número de cocinero asignado.
- Finalizar el preparado de un plato indicando el número de cocinero.
- Consultar la cantidad de platos preparados por cocinero y por el total de la cocina.
- Consultar el cocinero que más platos preparo.
- Consultar el promedio de platos por cocinero de la cocina.

3. Implementar el método buscarAtracciones de la clase AgenteDeViajes a partir de las siguientes especificaciones:

```
class AgenteDeViajes {

 public:
  /* post: busca en 'lugaresDisponibles' aquellas Atracciones que no estén en nombresDeLugaresYaVisitados, tengan lugares disponibles para cantidadDeVisitantes y cuyo valor total
   *      (la suma de los valores de cada atracción multiplicado por los visitantes) no supere 'dineroDisponibles'.
   */
  Lista<Atraccion*>* buscarAtracciones(Lista<Lugar*>* lugaresDisponibles,
                                       Lista<string>* nombresDeLugaresYaVisitados, unsigned int cantidadDeVisitantes, double dineroDisponibles);
};
```

```
class Lugar {

 public:

  /* post: Lugar identificado por 'nombre' y sin Atracciones.
   */
  Lugar(string nombre);

  /* post: devuelve el nombre del Lugar.
   */
  string obtenerNombre();

  /* post: devuelve todas las Atracciones que tiene.
   */
  Lista<Atraccion*>* obtenerAtracciones();

  /* post: libera todas las Atracciones del Lugar.
   */
  ~Lugar();
};
```

```
class Atraccion {

 public:

  /* post: Atracción identificada por 'nombre',
   *      cuya duración es 'minutos'
   */
  Atraccion(string nombre, unsigned int lugaresDisponibles, double valor)

  /* post: devuelve el nombre de la atracción. */
  string obtenerNombre();

  /* post: devuelve la cantidad de lugares disponibles. */
  unsigned int lugaresDisponibles();
  /* post: devuelve el valor por visitante de la atracción */
  double valor();

}
```

Para aprobar es necesario tener al menos el 60% de cada uno de los ejercicios correctos y completos.
Duración del examen : 3 horas

0
67666766
ACA
CBC
A65A
1
68676867
ACA
CBC
B66B
2
69686968
ACA
CBC
C67C
3
70697069
ACA
CBC
D68D
4
71707170
ACA
CBC
E69E
5
72717271
ACA
CBC
F70F
6
73727372
ACA
CBC
G71G
7
74737473
ACA
CBC
H72H
8
75747574
ACA
CBC
I73I
9
76757675
ACA
CBC
J74J
TODOS
delete A
delete[] E

Para aprobar es necesario tener al menos el 60% de cada uno de los ejercicios correctos y completos.
Duración del examen : 3 horas