Parcial Algoritmos y Programación II - 7541 Curso Calvo (2º instancia)
16 de Noviembre 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;
  int **B;
  char *D, *E;
  char G;
  int H;

  G = 'B';
  D = &G;
  E = D;
  (*E) = 'C';
  cout << (*D) << G << (*E) << endl;

  H = 64 + ULTIMO DIGITO PADRON;
  A = new int[2];
  (*A) = H;
  B = &A;
  C = A + 1;
  F = A;
  F[1] = 65;
  cout << (**B) << (*F) << (*C) << endl;

  E = (char*) (F + 1);
  G = 'D';
  (*C) = 66;
  cout << (*D) << (*E) << F[1] << endl;

  D = (char*) (*B);
  A[1] = 67;
  G = 'X';
  cout << (*E) << (*D) << (*C) << endl;

  // liberar la memoria
  // ...

  return 0;
}
```

2. Diseñar la especificación e implementar el TDA **Entrenamiento**. Un Entrenamiento se debe crear recibiendo como parámetro la cantidad de días de duración. Debe proveer operaciones para:

- devolver la duración (en días)
- registrar la cantidad de kilómetros corridos (por ejemplo 1.5) en uno de los días (identificado por número de orden).
- anular el entrenamiento en uno de los días.
- devolver la cantidad de kilómetros corridos en un día.
- devolver la cantidad de kilómetros corridos en total.
- devolver el promedio de kilómetros corridos por día (considerando solo aquellos días en los no fue anulado).
- devolver la cantidad de días que corrió más que el promedio.
- Devolver el dia que más kilómetros se corrió.

Nota: No debe ser posible registrar más de una vez la cantidad de kilómetros corridos en un día particular.

3. Implementar el método buscarReceteasPosibles de la clase Chef a partir de las siguientes especificaciones:

```
class Chef {
 public:
  /* post: busca en la lista recetas aquellas Recetas que pueden ser preparadas con los Ingredientes que se encuentran
   *      en ingredientesDisponibles y tengan mas ingredientes que cantidadDeIngredientesMinima. Considera que todos los Ingredientes de la Receta deben estar en
   *      ingredientesDisponibles comparando su nombre y validando que la cantidad sea la suficiente.
   *      Devuelve una nueva Lista con todos las Recetas en esta condición. */
  Lista<Receta*>* buscarRecetasPosibles(Lista<Receta*>* recetas, Lista<Ingrediente*>* ingredientesDisponibles, int cantidadDeIngredientesMinima);
};
```

```
class Receta {

 public:

  /* post: inicializa la Receta sin Ingredientes asignados.
   */
  Receta(string nombre);

  /* post: elimina todos los Ingrediente de la Receta.
   */
  ~Receta();

  /* post: devuelve todos los Ingredientes de la Receta.
   */
  Lista<Ingrediente*>* getIngredientes();

  /* post: devuelve el nombre que identifica a la Receta.
   */
  string getNombre();
};
```

```
class Ingrediente {

 public:

  /* post: queda inicializado con el nombre y cantidad
   *      indicados.
   */
  Ingrediente(string nombre, float cantidad);

  ~Ingrediente();

  /* post: devuelve el nombre del Ingrediente.
   */
  string getNombre();

  /* post: devuelve la cantidad del Ingrediente.
   */
  float getCantidad();
};
```

Los alumnos que tienen aprobado algún punto no deben realizarlo.
Para aprobar es necesario tener al menos el 60% de cada uno de los ejercicios correctos y completos.
Para cada método escribir pre y post condición, si recibe argumentos y cuáles, y si retorna un dato y cuál. De faltar esto, se considerará el código incompleto.
Duración del examen: 3 horas