

LENGUAJES DE PROGRAMACIÓN

Trabajo Práctico - Septiembre de 2020

INSTRUCCIONES

- El trabajo práctico debe realizarse de manera individual. No debe realizarse en grupo. Se penalizará cualquier uso compartido de las soluciones propuestas y de los códigos programados.
- El trabajo debe entregarse a través del curso virtual de la asignatura en la plataforma Alf.
- La fecha límite de entrega es el día 10 de septiembre.
- El alumno debe entregar un fichero comprimido, en formato zip o tar, que contenga:
 - o Una *memoria* en la cual explique la solución a los ejercicios, incluyendo los listados documentados del código C++ desarrollado y una descripción detallada de las pruebas realizadas al código para comprobar que funciona correctamente. Este documento deberá estar en formato pdf.
 - o Los ficheros del código fuente C++ solución a los ejercicios.

No deben entregarse ficheros ejecutables.

El nombre del fichero comprimido debe ser la concatenación de los apellidos y el nombre del alumno. Por ejemplo, GomezMartinLuisa.zip

CRITERIOS DE EVALUACIÓN

- Para que el trabajo pueda ser corregido, es imprescindible que el alumno entregue dentro del plazo establecido un fichero comprimido que contenga la memoria en formato pdf y el código fuente C++ de los ejercicios que haya realizado.
- El trabajo se compone de 4 ejercicios, cada uno de los cuales se valorará sobre 2.5 puntos.
- Para aprobar el trabajo es necesario que la nota total obtenida en los ejercicios sea mayor o igual que 5.
- Si el código solución de un ejercicio tiene errores de compilación o no tiene la funcionalidad pedida, dicho ejercicio se valorará con cero puntos.
- Si el código solución de un ejercicio compila sin errores y tiene la funcionalidad pedida, la puntuación en dicho ejercicio será al menos de 2 puntos.
- Se valorará positivamente la eficiencia y la adecuada documentación del código, así como la presentación y calidad de las explicaciones proporcionadas en la memoria.

CONTENIDO DE LA MEMORIA

Debe incluirse en la memoria, para cada uno de los ejercicios:

1. Listado del código fuente debidamente documentado.
2. Enumeración del conjunto completo de pruebas que usted ha realizado al código para comprobar que funciona correctamente.
3. Un ejemplo de ejecución del código, incluyendo capturas de pantalla en las que puedan verse las distintas fases en la ejecución del programa, tales como la entrada de los datos, la salida de los resultados, la escritura en la consola de mensajes de error, etc. según proceda en cada caso.

EJERCICIO 1

Consideremos el polinomio de grado n siguiente

$$p_n(x) = a_0 + a_1 \cdot x + \cdots + a_n \cdot x^n$$

cuyos coeficientes a_0, a_1, \dots, a_n son números reales.

Escriba un programa en C++ que permita al usuario especificar el orden y el valor de los coeficientes del polinomio, y que calcule y escriba en la consola los valores obtenidos de evaluar el polinomio para un determinado conjunto de valores de x .

El programa debe realizar las acciones siguientes:

1. Escribir un mensaje en la consola indicando al usuario que introduzca el orden del polinomio.
2. Leer el valor introducido por el usuario, almacenándolo en una variable de tipo entero llamada n . Si el valor de n es menor que cero, terminar.
3. Escribir un mensaje en la consola solicitando al usuario que introduzca por consola el valor de los $n + 1$ coeficientes a_0, a_1, \dots, a_n . Los valores deben ser leídos de la consola y almacenados en un vector de tipo **double** llamado `coefs`.
4. Calcular y escribir en la consola, en formato científico con 8 dígitos de precisión, el valor del polinomio para cada uno de los valores de x siguientes:

$$x_k = \cos\left(\frac{2 \cdot \pi \cdot k}{N}\right) \quad \text{con } k = 0, \dots, N - 1$$

donde N es una constante global del programa cuyo valor es 10.

5. Terminar.

EJERCICIO 2

Consideremos una magnitud física escalar (por ejemplo, la temperatura), que llamaremos U , dependiente de la coordenada espacial x y del tiempo t . El valor de U en cada punto (x, t) , tal que $0 < x < 1$ y $t > 0$, puede calcularse resolviendo la ecuación de difusión en una dimensión:

$$\frac{\partial^2 U}{\partial x^2} = \frac{\partial U}{\partial t}, \quad 0 < x < 1, \quad t > 0$$

con las condiciones de contorno:

$$U(0, t) = 0 = U(1, t), \quad t > 0$$

y la condición inicial:

$$U(x, 0) = 100, \quad 0 < x < 1$$

Nuestro objetivo es estimar el valor de U en un determinado punto (x^*, t^*) . Para ello, discretizamos la coordenada espacial en el intervalo $[0, 1]$:

$$x_i = \frac{i}{I} \quad \text{con} \quad i = 0, 1, \dots, I$$

donde I es un número natural, y discretizamos el tiempo:

$$t_n = n \cdot \Delta t \quad \text{con} \quad n = 0, 1, \dots$$

donde el paso de avance en el tiempo, Δt , es un número real mayor que cero.

El punto (x^*, t^*) debe ser uno de los puntos de la discretización:

$$x^* = \frac{i^*}{I}, \quad t^* = n^* \cdot \Delta t$$

donde i^* y n^* son números enteros mayores que cero, y además de satisface $i^* < I$.

El algoritmo que emplearemos para estimar $U(x^*, t^*)$ es un método de Monte Carlo. Consiste en generar gran número de caminos aleatorios en el espacio bidimensional $x-t$ discretizado. Cada camino aleatorio comienza en (x^*, t^*) y termina cuando alcanza un punto cualquiera del contorno (x vale 0 ó 1) o del instante inicial (t vale 0). El algoritmo contabiliza el valor de U en el punto terminal del camino, ya que la media de dichos valores es una estimación de $U(x^*, t^*)$. El algoritmo es descrito a continuación.

Algoritmo

1. Seleccionar el punto (x^*, t^*) .
2. Discretizar la coordenada espacial y el tiempo:
 - a) Asignar un valor entero positivo a la constante I .
 - b) Asignar

$$\Delta t = \frac{1}{2 \cdot I^2}$$

Obsérvese que de esta manera se satisface $\frac{(\Delta x)^2}{\Delta t} = 2$.

3. Asignar valor a N_C . Esta constante entera es el número de caminos aleatorios que recorrerá el algoritmo.
4. Asignar $k = 1$.
5. Situarse en el punto inicial del camino: $(x, t) = (x^*, t^*)$.
6. Avanzar un paso en el camino. Para ello, generar un número pseudoaleatorio, r , y emplearlo para decidir a qué punto nos movemos:

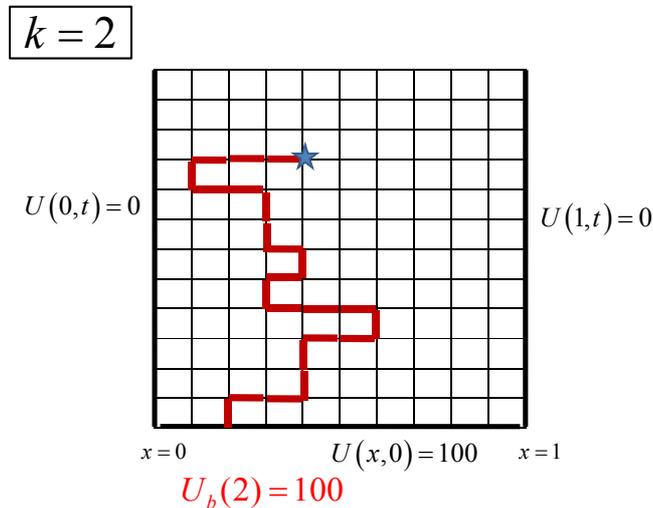
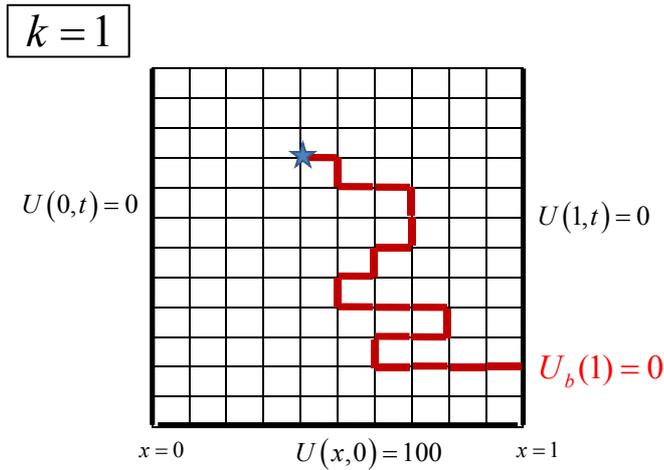
$$\begin{aligned} &\text{de } (x, t) \text{ a } (x + \Delta x, t) \text{ si } 0 \leq r < 0.25 \\ &\text{de } (x, t) \text{ a } (x - \Delta x, t) \text{ si } 0.25 \leq r < 0.5 \\ &\text{de } (x, t) \text{ a } (x, t - \Delta t) \text{ si } 0.5 \leq r \leq 1 \end{aligned}$$

Obsérvese que el avance en el tiempo, de producirse, siempre es hacia atrás, nunca hacia adelante.

7. Si el punto al que nos hemos movido no pertenece ni al contorno, ni al instante inicial, ir al Paso 6 del algoritmo.
8. Asignar a $U_b(k)$ el valor que tiene U en el punto del contorno o inicial que hemos alcanzado en este camino aleatorio k -ésimo.
9. Si $k < N_C$, asignar $k = k + 1$ e ir al Paso 5 del algoritmo.
10. Calcular $U(x^*, t^*)$ de la forma siguiente:

$$U(x^*, t^*) = \frac{1}{N_C} \cdot \sum_{k=1}^{N_C} U_b(k)$$

11. Terminar.



Como ilustración, sobre estas líneas se muestran dos caminos aleatorios. Obsérvese que cada camino aleatorio es una sucesión de nodos de la cuadrícula obtenida de discretizar la coordenada espacial (eje horizontal) y el tiempo (eje vertical, creciente hacia arriba). Todo camino comienza en el punto (x^*, t^*) , que está señalado mediante una estrella en la figura, y termina cuando alcanza el contorno (como en el camino $k = 1$) o el instante inicial (como en el camino $k = 2$). El valor terminal del camino $k = 1$ es $U_b(1) = 0$ y el del camino $k = 2$ es $U_b(2) = 100$.

Escriba un programa en C++ que estime el valor de U empleando el algoritmo anterior y escriba dicho valor en la consola. Ejecútelo para estimar, empleando $I = 200$ y $N_C = 10000$, el valor de U en los dos puntos siguientes:

$$(x^* = 0.4, t^* = 0.04)$$

$$(x^* = 0.4, t^* = 0.1)$$

EJERCICIO 3

El método de Newton de las diferencias divididas es una técnica de interpolación polinómica. Se describe a continuación.

Supongamos que conocemos el valor de una función $f(x)$ en $n + 1$ puntos diferentes.

$$\begin{array}{ll} x_0 & f(x_0) \\ x_1 & f(x_1) \\ \vdots & \\ x_i & f(x_i) \\ \vdots & \\ x_n & f(x_n) \end{array}$$

El polinomio de interpolación de grado n es el siguiente

$$\begin{aligned} p_n(x) = & f[x_0] \\ & + (x - x_0) \cdot f[x_1, x_0] \\ & + (x - x_0) \cdot (x - x_1) \cdot f[x_2, x_1, x_0] \\ & \dots \\ & + (x - x_0) \cdot (x - x_1) \cdot (x - x_2) \cdot \dots \cdot (x - x_{n-1}) \cdot f[x_n, x_{n-1}, \dots, x_0] \end{aligned}$$

donde las diferencias divididas $f[\dots]$ se calculan de la forma indicada a continuación

Orden	Notación	Definición
0	$f[x_0]$	$f(x_0)$
1	$f[x_1, x_0]$	$\frac{f[x_1] - f[x_0]}{x_1 - x_0}$
2	$f[x_2, x_1, x_0]$	$\frac{f[x_2, x_1] - f[x_1, x_0]}{x_2 - x_0}$
3	$f[x_3, x_2, x_1, x_0]$	$\frac{f[x_3, x_2, x_1] - f[x_2, x_1, x_0]}{x_3 - x_0}$
	\vdots	
n	$f[x_n, x_{n-1}, \dots, x_0]$	$\frac{f[x_n, x_{n-1}, \dots, x_1] - f[x_{n-1}, x_{n-2}, \dots, x_0]}{x_n - x_0}$

A continuación se muestra un ejemplo. Supongamos que conocemos el valor de una función $f(x)$ en 4 puntos diferentes.

i	x_i	$f(x_i)$
0	0	-5
1	1	1
2	3	25
3	4	55

Se desea calcular el polinomio de interpolación de grado 3. Para ello, calculamos las diferencias divididas.

$$\begin{aligned}
 f[x_0] &= f(x_0) &= -5 \\
 f[x_1] &= f(x_1) &= 1 \\
 f[x_2] &= f(x_2) &= 25 \\
 f[x_3] &= f(x_3) &= 55 \\
 f[x_3, x_2] &= \frac{f[x_3]-f[x_2]}{x_3-x_2} = \frac{55-25}{3-2} &= 30 \\
 f[x_2, x_1] &= \frac{f[x_2]-f[x_1]}{x_2-x_1} = \frac{25-1}{3-1} &= 12 \\
 f[x_1, x_0] &= \frac{f[x_1]-f[x_0]}{x_1-x_0} = \frac{1-(-5)}{1-0} &= 6 \\
 f[x_3, x_2, x_1] &= \frac{f[x_3, x_2]-f[x_2, x_1]}{x_3-x_1} = \frac{30-12}{4-1} &= 6 \\
 f[x_2, x_1, x_0] &= \frac{f[x_2, x_1]-f[x_1, x_0]}{x_2-x_0} = \frac{12-6}{3-0} &= 2 \\
 f[x_3, x_2, x_1, x_0] &= \frac{f[x_3, x_2, x_1]-f[x_2, x_1, x_0]}{x_3-x_0} = \frac{6-2}{4-0} &= 1
 \end{aligned}$$

El polinomio de interpolación de grado 3 se muestra a continuación.

$$\begin{aligned}
 p_3(x) &= f[x_0] \\
 &+ (x - x_0) \cdot f[x_1, x_0] \\
 &+ (x - x_0) \cdot (x - x_1) \cdot f[x_2, x_1, x_0] \\
 &+ (x - x_0) \cdot (x - x_1) \cdot (x - x_2) \cdot f[x_3, x_2, x_1, x_0] \\
 &= -5 + x \cdot 6 + x \cdot (x - 1) \cdot 2 + x \cdot (x - 1) \cdot (x - 3)
 \end{aligned}$$

La interpolación para un determinado valor de x se calcula sustituyendo dicho valor en el polinomio. Por ejemplo, el valor interpolado en $x = 0.5$ es

$$\begin{aligned}
 p_3(0.5) &= -5 + 0.5 \cdot 6 + 0.5 \cdot (0.5 - 1) \cdot 2 + 0.5 \cdot (0.5 - 1) \cdot (0.5 - 3) \\
 &= -1.875
 \end{aligned}$$

Escriba un programa en C++ que realice las acciones siguientes.

1. Abrir para lectura un fichero de texto llamado `puntos.txt`. Si se produce error, terminar.
2. Leer el contenido del fichero `puntos.txt`. En este fichero están escritos, en dos columnas, los valores x_i (primera columna) y $f(x_i)$ (segunda columna). Estos valores son, en general, números reales.

El fichero `puntos.txt` correspondiente al ejemplo anterior es:

```
0  -5
1  1
3  25
4  55
```

Los valores leídos deben almacenarse en un vector de estructuras `Punto`, la cual debe definirse de la forma siguiente:

```
struct Punto {
    double x;
    double fx;
};
```

3. Si el número de puntos es menor que dos, o si alguno de los puntos está repetido, mostrar un mensaje indicándolo y terminar.
4. Solicitar al usuario, mediante un mensaje en la consola, que introduzca el valor de x para el cual quiere evaluar el polinomio de interpolación.
5. Si el valor introducido es mayor que todos los valores x_i leídos del fichero o si es menor que todos ellos, mostrar un mensaje en la consola indicándolo y terminar.
6. Mostrar en la consola el valor obtenido de sustituir x en el polinomio de interpolación de grado n calculado de los valores leídos del fichero, donde el número de valores leídos es $n + 1$.
7. Terminar.

Muestre en la memoria el resultado obtenido al ejecutar su programa para cada uno de los dos ficheros `puntos.txt` siguientes, interpolando en $x = 0.5$.

Primer fichero de prueba

```
0 -5
1 1
3 25
4 55
```

Segundo fichero de prueba

```
0.0 1.000
0.2 0.980
0.3 0.956
0.4 0.921
0.6 0.825
0.7 0.765
0.9 0.622
1.0 0.540
```

EJERCICIO 4

Un grafo $G = (V, E)$ consiste en un conjunto de vértices $V = \{v_1, v_2, \dots\}$ y un conjunto de arcos $E = \{e_1, e_2, \dots\}$, tal que cada arco es identificado mediante una pareja no ordenada (v_i, v_j) de vértices. La forma más común de representar un grafo es mediante un diagrama, en el cual cada vértice es representado mediante un punto y cada arco mediante un segmento de línea conectando sus vértices.

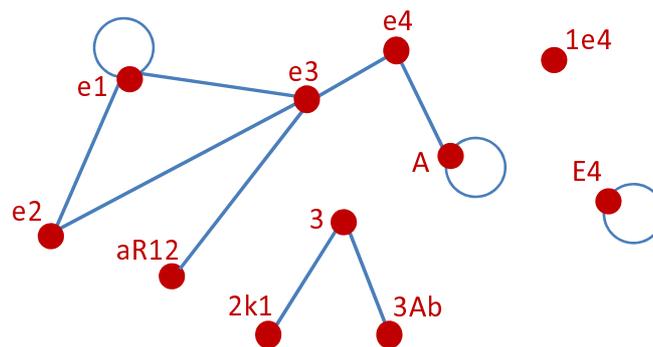
Supongamos que se emplean dos ficheros de texto para describir el grafo:

- En el primero, llamado `vertices.txt`, están escritos en columna los identificadores de los vértices. El identificador de un vértice es una cadena de caracteres compuesta por letras y/o números. El identificador de cada vértice debe ser único, es decir, no puede haber dos vértices con el mismo identificador.
- En el segundo, llamado `arcos.txt`, están escritas las parejas de vértices de cada arco. Cada pareja de vértices está escrita en una línea del fichero.

Escriba un programa en C++ que realice las acciones siguientes:

1. Leer los dos ficheros de texto `vertices.txt` y `arcos.txt`. Si se produce error, indicarlo mediante un mensaje en la consola y terminar.
2. Comprobar que ninguno de los identificadores del fichero `vertices.txt` está repetido. Si alguno está repetido, indicarlo mediante un mensaje en la consola y terminar.
3. Comprobar que todos los identificadores que aparecen en `arcos.txt`, aparecen en el fichero `vertices.txt`. Si no es así, indicarlo mediante un mensaje en la consola y terminar.
4. Escribir en consola, para cada vértice del grafo, qué vértices están directamente conectados a él mediante un arco. La información de cada vértice debe escribirse en una línea, con este formato: identificador del vértice, símbolo dos puntos (:), identificadores de los vértices directamente conectados.
5. Terminar.

Como ejemplo, a continuación se muestra el diagrama de un grafo. Junto a cada vértice se ha escrito su identificador. Debajo del diagrama, se muestra una forma de describir el grafo empleando los dos ficheros de texto. Finalmente se muestra la salida por consola del programa.



Fichero vertices.txt

```
e3
E4
e2
e4
A
1e4
3Ab
2k1
e1
3
aR12
```

Fichero arcos.txt

```
e1 e2
e1 e1
e1 e3
e3 e4
e2 e3
aR12 e3
A e4
E4 E4
A A
3 2k1
3Ab 3
```

Salida por consola del programa

```
1e4 :
2k1 : 3
3 : 2k1 3Ab
3Ab : 3
A : e4 A
E4 : E4
aR12 : e3
e1 : e2 e1 e3
e2 : e1 e3
e3 : e1 e4 e2 aR12
e4 : e3 A
```