

LENGUAJES DE PROGRAMACIÓN

Trabajo Práctico - Septiembre de 2023

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:
 - 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.
 - Los ficheros del código fuente C++ (ficheros de texto con extensión cpp) 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++ (ficheros de texto con extensión cpp) 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. Se recomienda comprobar que el programa compila y ejecuta correctamente con el compilador online siguiente: https://www.onlinegdb.com/online_cplusplus_compiler
- 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. Descripción de las pruebas que usted ha realizado al código para comprobar que funciona correctamente.
3. Ejemplos 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

La viscosidad η de un fluido de densidad ρ_f puede determinarse midiendo la velocidad límite v que adquiere una esfera de masa m y radio R que cae libremente completamente sumergida en el fluido (en determinadas condiciones experimentales que no vienen al caso). La relación entre las magnitudes físicas anteriores es:

$$v = \frac{2 \cdot g \cdot (\rho_e - \rho_f) \cdot R^2}{9 \cdot \eta}$$

donde g es la aceleración gravitatoria. La densidad de la esfera ρ_e vale:

$$\rho_e = \frac{m}{\frac{4}{3} \cdot \pi \cdot R^3}$$

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

1. Declarar una constante de tipo real llamada g (que representa la aceleración de la gravedad) y asignarle el valor 9.8.
2. Escribir mensajes en la consola solicitando al usuario que introduzca los valores de las magnitudes físicas siguientes: la densidad del fluido, y la masa y el radio de la esfera. Leer dichos valores, almacenándolos en sendas variables reales llamadas `rho_f`, `m` y `R`.
3. Calcular la densidad de la esfera y escribirla en la consola. Si ésta es menor o igual que la densidad del fluido, mostrar un mensaje en la consola indicándolo y terminar.
4. Escribir un mensaje en la consola solicitando al usuario que escoja una de las dos opciones siguientes: (1) Calcular la velocidad límite; y (2) Calcular la viscosidad del fluido.
5. Leer el valor introducido por consola, almacenándolo en una variable entera llamada `opcion`. Si el valor es diferente de 1 y de 2, terminar.
6. Si el usuario ha seleccionado la opción 1, solicitarle que introduzca por consola el valor de la viscosidad del fluido, leer el valor, y calcular y escribir en la consola la velocidad límite. Si el usuario ha seleccionado la opción 2, solicitarle que introduzca por consola la velocidad límite, leer el valor, calcular y escribir en la consola la viscosidad del fluido.
7. Terminar.

Los valores calculados deben escribirse en la consola en formato fijo, con 3 dígitos de precisión.

Puede asumir que el usuario va a introducir las magnitudes físicas expresadas en unas unidades tales que las ecuaciones mostradas en el enunciado sean correctas dimensionalmente.

Muestre en la memoria el resultado obtenido al ejecutar su programa para los casos de prueba siguientes.

- Caso de prueba 1. $\rho_f = 1260$, $m = 0.038$, $R = 0.01$, Opción 1, $\eta = 1.390$.
- Caso de prueba 2. $\rho_f = 1260$, $m = 0.038$, $R = 0.01$, Opción 2, $v = 1.224$.
- Caso de prueba 3. $\rho_f = 1260$, $m = 0.46$, $R = 0.05$, Opción 1, $\eta = 1.390$.

Ejercicio 2

Una partícula puede encontrarse en cualquiera de los cinco estados siguientes: C (center), U (up), D (down), L (left) y R (right). La secuencia de estados en que se ha encontrado la partícula durante un cierto periodo de tiempo está almacenada en un fichero de texto llamado *estados.txt*.

A continuación se muestra un ejemplo del contenido del fichero *estados.txt*. Obsérvese que los estados están separados mediante espacio/s en blanco y/o salto de línea.

```
C U D L C D L R D L C R L C D R L C R D L C R U C D U D U C
R L D C U D C R C D R D R C D L C L R D L U D L D U D U D U
D L D C R C L C D U D U C D R L R L C R D L C R C U C D L R
D C R C D R D C D L C L R D L D L C D L R D L D C D R L U D
```

Se trata de calcular el número de transiciones al estado C que se ha producido desde cada uno de los otros 4 estados (U, D, L, R). Para ello, escriba un programa en C++ que realice las acciones siguientes:

1. Abrir el fichero *estados.txt* para lectura. Si se produce error, escribir un mensaje en la consola indicándolo y terminar.
2. Leer el contenido del fichero, almacenando su contenido en un vector de tipo **char**.
3. Realizar las dos comprobaciones siguientes:
 - a) El fichero no contiene ningún carácter diferente a C, U, D, L, R.
 - b) El fichero no contiene dos caracteres seguidos iguales.

Si cualquiera de estas comprobaciones falla, mostrar un mensaje en la consola indicándolo y terminar.

4. Calcular el número de transiciones al estado C producidas desde cada uno de los otros 4 estados (U, D, L, R) y escribir dichos números en la consola.
5. Terminar.

Muestre en la memoria el resultado obtenido al ejecutar su programa empleando el fichero *estados.txt* proporcionado como ejemplo en el enunciado.

Ejercicio 3

La evolución en el tiempo de la variable y está descrita mediante la ecuación diferencial ordinaria

$$\frac{dy}{dt} = f(y) \quad \text{con} \quad f(y) = y^3 \cdot \exp\left(\frac{1}{y} - y^6\right)$$

donde t representa el tiempo. El valor de y en el instante $t = 0$ es

$$y(t = 0) = y_0$$

siendo y_0 un valor real conocido.

Se desea calcular el valor de y en los instantes de tiempo

$$t_0 = 0, \quad t_1 = h, \quad t_2 = 2 \cdot h, \quad \dots, \quad t_n = n \cdot h, \quad \dots, \quad t_N = N \cdot h$$

donde la constante h , denominada “longitud del paso de integración”, tiene un valor conocido. La constante de valor entero N también es conocida.

El valor y_{n+1} (valor de la variable y en el instante t_{n+1}) se calcula a partir y_n (valor de la variable y en el instante t_n) de la forma siguiente:

$$\begin{aligned} k_1 &= \frac{h}{s} \cdot f(y_n) \\ k_2 &= \frac{h}{s-1} \cdot f(y_n + k_1) \\ k_3 &= \frac{h}{s-2} \cdot f(y_n + k_2) \\ &\dots \\ y_{n+1} &= y_n + h \cdot f(y_n + k_{s-1}) \end{aligned}$$

para $n = 0, \dots, N - 1$, donde s es una constante conocida, de valor entero, que determina el orden del método numérico de integración. Es decir:

– En el caso $s = 2$,

$$\begin{aligned} k_1 &= \frac{h}{2} \cdot f(y_n) \\ y_{n+1} &= y_n + h \cdot f(y_n + k_1) \end{aligned}$$

– En el caso $s = 3$,

$$\begin{aligned}k_1 &= \frac{h}{3} \cdot f(y_n) \\k_2 &= \frac{h}{2} \cdot f(y_n + k_1) \\y_{n+1} &= y_n + h \cdot f(y_n + k_2)\end{aligned}$$

– En el caso $s = 4$,

$$\begin{aligned}k_1 &= \frac{h}{4} \cdot f(y_n) \\k_2 &= \frac{h}{3} \cdot f(y_n + k_1) \\k_3 &= \frac{h}{2} \cdot f(y_n + k_2) \\y_{n+1} &= y_n + h \cdot f(y_n + k_3)\end{aligned}$$

– ... y así sucesivamente.

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

1. Declarar la constante de tipo entero llamada N y asignarle el valor 10^4 .
2. Declarar una constante de tipo real llamada h y asignarle el valor 10^{-4} .
3. Mediante un mensaje escrito en la consola, solicitar al usuario que escriba el valor de y_0 . Leer el valor introducido por el usuario, almacenándolo en una variable de tipo real llamada $y0$.
4. Mediante un mensaje escrito en la consola, solicitar al usuario que escriba el valor de s . Debe ser un número entero mayor que uno. Leer el valor introducido por el usuario, almacenándolo en una variable de tipo entero llamada s . Si no se satisface que s es mayor que uno, el programa debe terminar.
5. Calcular y_1, \dots, y_N para los valores de y_0 y s introducidos por el usuario.
6. Escribir en la consola el valor y_N , en formato fijo con una precisión de 10 dígitos.
7. Terminar.

Muestre en la memoria el resultado obtenido al ejecutar su programa para cada uno de los dos casos de prueba siguientes. Caso de Prueba 1: $y_0 = 0.1$, $s = 3$. Caso de Prueba 2: $y_0 = 1$, $s = 10$.

Ejercicio 4

En base a las calorías y el precio por ración de una serie de platos de comida y de postres, se plantea el problema de elaborar menús escolares que cumplan las condiciones siguientes:

- Un menú debe estar compuesto por un primer plato, un segundo plato y un postre.
- El menú en su conjunto debe tener un número de calorías comprendido en un intervalo $[m, M]$, siendo m y M números enteros conocidos tales que $0 < m < M$.
- El precio del menú debe ser no superior a P euros, siendo P un valor real conocido.

La información de los posibles platos de comida y postres se encuentra almacenada en un fichero de texto llamado *carta.txt*, en el formato descrito a continuación. En cada línea del fichero se encuentra descrito un plato o postre. En la primera columna se indica su nombre. En la segunda columna se indica mediante un número entero si se trata de un primer plato (1), un segundo plato (2) o un postre (3). En la tercera columna se indica el número de calorías por ración, el cual es un número entero. Finalmente, en la cuarta columna se indica el precio en euros de la ración de dicho plato o postre. En el fichero, los platos no están ordenados y no hay platos repetidos.

A continuación se muestra un ejemplo de fichero *carta.txt*.

Merluza	2	161	3.20
Flan	3	137	0.85
San_jacobo	2	350	1.38
Paella	1	379	2.78
Lentejas	1	412	2.05
Cinta_de_lomo_plancha	2	220	2.30
Natillas	3	95	0.60
Albondigas_ternera	2	283	3.50
Tortilla_patata	2	320	2.60
Crema_de_calabacin	1	169	1.80
Helado	3	120	0.40
Cordero_asado_con_patatas	2	380	3.20
Boquerones	2	234	2.00
Pechuga_de_pollo_empanada	2	327	2.60
Fabada_asturiana	1	426	2.60
Judias_verdes	1	282	1.90
Naranja	3	38	0.30
Gelatina	3	39	0.15
Ensalada_mixta	1	227	2.40

Fresas	3	29	1.20
Gazpacho	1	121	0.90
Tarta_de_queso	3	203	1.20
Bacaladitos_con_ensalada	2	200	2.05
Manzana	3	72	0.30
Lasagna	2	378	2.60
Sopa_de_pescado	1	290	1.50
Filete_ternera	2	190	3.80
Chuleta_de_cerdo	2	389	2.30
Huevos_relLENOS	1	221	1.20
Macarrones_con_tomate_y_carne_picada	1	569	1.50
Bacalao_con_patatas	2	465	2.90
Platano	3	89	0.95

Así, por ejemplo, un menú compuesto por gazpacho, albondigas_ternera y natillas tendría $121+283+95 = 499$ calorías en total y su precio total sería $0.90+3.50+0.60 = 5.0$ euros.

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

1. Escribir un mensaje en la consola solicitando al usuario que introduzca los valores de m y M (valores mínimo y máximo de calorías por menú), y el precio máximo P .
2. Leer los valores introducidos por consola, almacenándolos en las variables enteras m y M , y en la variable real P . Si no se satisface $0 < m < M$, o no se satisface $P > 0$, mostrar un mensaje en la consola indicándolo y terminar.
3. Leer el fichero *carta.txt*, almacenando su contenido en una o varias estructuras de datos. Puede asumir que el formato del fichero es correcto.
4. Escribir en la consola todos los menús que satisfacen simultáneamente estas dos condiciones: su número de calorías está en el intervalo $[m, M]$, y su precio es menor o igual que P .

Los menús deben escribirse ordenados de menor a mayor precio, especificando el número de calorías total y precio total de cada uno.

5. Terminar.

Muestre en la memoria el resultado de ejecutar su programa empleando el fichero *carta.txt* listado anteriormente, y cada uno de los dos casos de prueba siguientes:

- Caso de Prueba 1: $m = 600$ calorías, $M = 750$ calorías y $P = 4$ euros.
- Caso de Prueba 2: $m = 450$ calorías, $M = 550$ calorías y $P = 5$ euros.