UNIVERSIDAD DE CARABOBO

Facultad Experimental de Ciencias y Tecnología

Departamento de Computación

Fundamentos de Programación

Cod. Asignatura: TAO207 **Modelo:** 1

Taller 1

Normas de Entrega:

- El nombre de cada archivo **.c**, archivo de entrada y archivo de salida debe coincidir con lo indicado por cada ejercicio.
- Debe colocar la siguiente información en la cabecera de cada archivo fuente por medio de un comentario de bloque: Fecha, Nombre, Apellido, Cédula, Número Sección y Número del Modelo del Parcial.
- La entrega de los programas se deberá realizar por medio de un archivo comprimido (**.zip**), el cual, debe contener una sub-directorio por cada ejercicio. Cada sub carpeta deberá tener el archivo fuente (**.c**) y la entrada del programa correspondiente.
- No adjunte los archivos de salida del programa, ni adjunte los ejecutables generados por el compilador.

No seguir las indicaciones antes mencionadas repercutirá en la nota final de su entrega.

IMPORTANTE

Deberá crear un menú sencillo que sea capaz de ejecutar el código correspondiente a la solución del ejercicio 1, así como la del ejercicio 2 dependiendo de la entrada del usuario. Esto implica que ambas soluciones pertenecerán al mismo archivo (**.c**). **(3 ptos)**

Este menú será de uso único, por lo cual si se ingresa una opción inválida, deberá finalizar la ejecución del programa sin mostrar ninguna de las dos soluciones posibles.

Ejemplos de uso

Entrada Terminal	**Salida Terminal**
1 …	Ejecución de ejercicio 1 …

Entrada Terminal	**Salida Terminal**
2 …	Ejecución de ejercicio 2 …

Entrada Terminal	**Salida Terminal**
3	Opción no reconocida

Ejercicio 1. (9 ptos)

Al llegar las vacaciones, se le ha ocurrido jugar el videojuego indie Noita por recomendación de un compañero de clases. Este juego trata de un pequeño hechicero que busca luchar contra el jefe final de una mazmorra. Al pasar el tiempo, ha logrado completar el juego y comienza a plantearse si hay alguna forma de hacer al juego más emocionante, y recuerda que el mismo posee una comunidad de modders (jugadores que crean contenido adicional para un juego existente) bastante activa.

Motivado por los esfuerzos de la comunidad, usted decide crear un Mod (componente personalizado de un juego) para añadir una serie de habilidades (Perks) que añaden tanto beneficios como desafíos adicionales al jugador.

Entre ellos, Se le ha ocurrido crear el Perk: *Ruleta espacial*, la cual potencia positiva o negativamente los puntos de ataque (d) del jugador en función de: Nivel del jugador (N), Los puntos de daño base que tiene la varita seleccionada actualmente (d_{base}) , y la posición en el eje X del jugado (x).

La idea del Perk consiste en que el daño de la varita pueda aumentar o disminuir cada vez que el jugador sale de ciertas regiones específicas del mapa, y que este efecto sea más fuerte a medida que el jugador suba de nivel.

Tal idea le ha permitido diseñar la siguiente fórmula que le permitirá calcular el daño total de la varita:

$$d = d_{base} + N * \frac{\sum_{i \geqslant 0}^{N} (sin(i+x))}{\prod_{j=1}^{4} 2}$$

Ahora, debido a que debe probar el comportamiento de la función de daño, también ha decidido probarla con M escenarios distintos. Cree un programa que le permita calcular el daño realizado por la varita en los M escenarios planteados.

Formato de Entrada

Cantidad_M_Escenarios Nivel_jugador_1 daño_base_1 posicion_x_1 Nivel_jugador_2 daño_base_2 posicion_x_2 ... Nivel_jugador_m daño_base_m posicion_x_m

Ejemplos de entrada y salida

Entrada Terminal	**Salida Terminal**
9	
1 2 3	Daño Realizado: 1.96
11 12 13	Daño Realizado: 12.14
14 16 20	Daño Realizado: 17.64
1000 35 3.14	Daño Realizado: -15.72
1000 35 1.5	Daño Realizado: 134.48
1000 70 0	Daño Realizado: 120.87
2000 70 0	Daño Realizado: 284.57
1 70 0	Daño Realizado: 70.05

Ejercicio 2 (8 ptos)

Se tiene una máquina que consiste en un conjunto de 6 estados (a, b, c, d, e, f) y un conjunto de 3 entradas (0, 1, 2). Esta máquina se inicia en un cierto **estado**, leído por teclado y finaliza en ese mismo u otro; también leído por teclado. Para ir de un estado al siguiente, la máquina se mueve por una serie de estados intermedios, que dependen de las entradas, es decir; para saber el estado siguiente en el cual estará la máquina, se debe proporcionar una entrada y en base a ésta, se conocerá el estado siguiente (Siempre es posible ir de un estado a otro). Dicha información se resume en la siguiente tabla de transiciones:

	Entradas		
Estados	**0**	**1**	**2**
A	B	A	D
B	C	E	F
C	A	C	D
D	D	A	F
E	B	E	F
F	F	D	F

Se desea crear un programa capaz de leer un estado inicial junto a una lista de N entradas y muestre el **estado final** del programa.

Formato de Entrada

```
Estado_Inicial N_Entradas
Entrada_1
Entrada_2
…
Entrada_N
```

Ejemplo de entrada y salida

Entrada Terminal	**Salida Terminal**
A 5 0 1 1 2 1	D