UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

Facultad de Ingeniería

Profesor: M.I. Oscar René Valdez Casillas

Fundamentos de Programación

Práctica de estudio 08: Estructuras de repetición

Integrantes:

Arteaga Gonzalez Jaime Alejandro

Márquez Alan

Valencia Francisco Valentín

García Jiménez Joel David

Grupo: 31

Arteaga, J., Márquez, A., García, J. y Valencia, V., (2024). Guía de práctica de estudio 08: Estructuras de repetición

Índice:

Resumen: Se realizaron dos ejercicios en C: el primero consiste en elaborar un programa que calcule el término 10 de la secuencia de Fibonacci, donde los dos primeros números son 0 y 1, y los restantes se obtienen sumando los dos anteriores. El segundo ejercicio implica el desarrollo de un programa para gestionar la venta de boletos en un estadio, que tiene 5 tipos de localidades identificadas por claves numéricas del 1 al 5, calculando los precios y manejando los datos de ventas para el próximo juego.

Introducción:

En esta práctica, se desarrollarán programas en C para resolver problemas básicos utilizando estructuras de repetición. Se comenzará con un programa que emplee la estructura while, seguido de otro que utilice do-while, permitiendo comparar sus diferencias. Además, se resolverá un problema asignado con la estructura for. Finalmente, se incorporará la directiva #define para crear código más versátil y reutilizable, fortaleciendo así las habilidades de programación.

Objetivo:

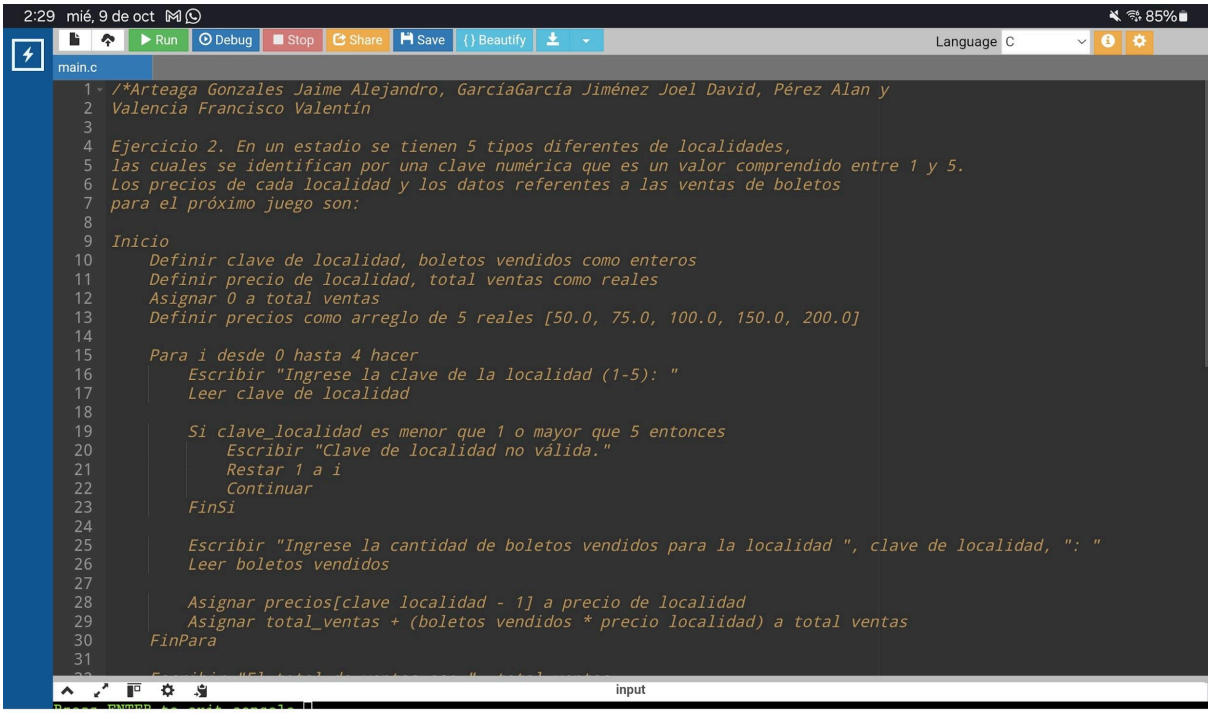
El alumno elaborará programas en C para la resolución de problemas básicos que incluyan las estructuras de repetición.

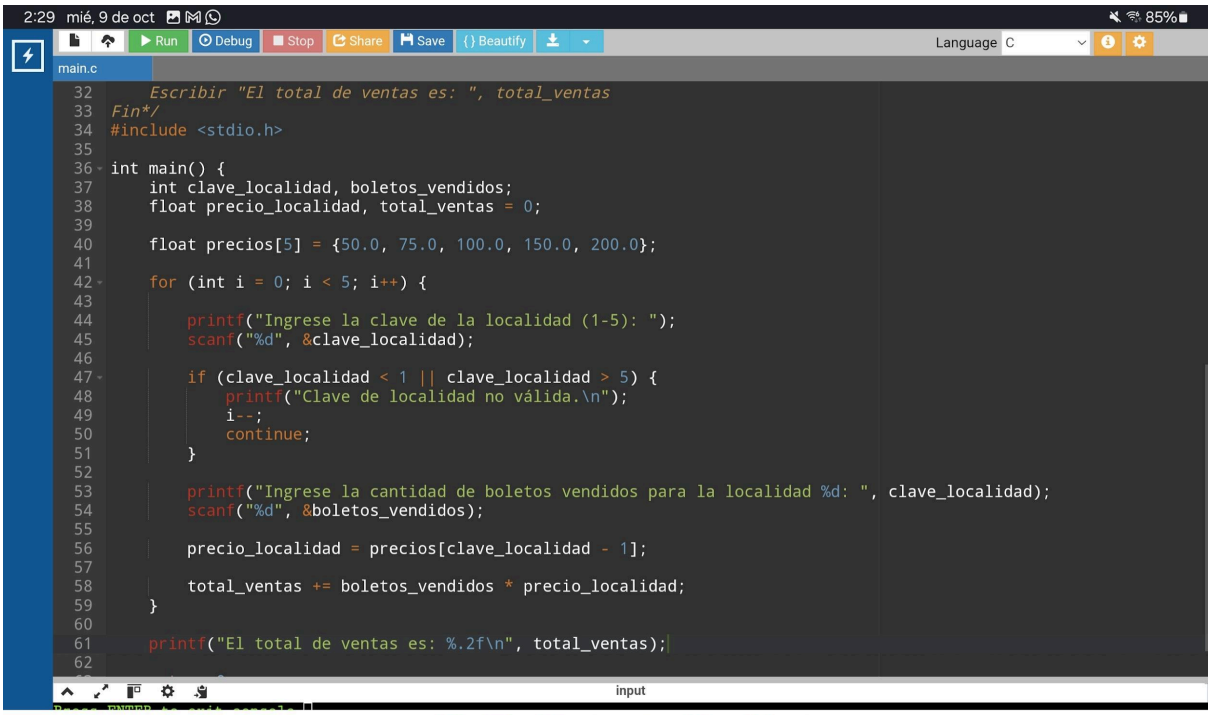
Ejercicios:

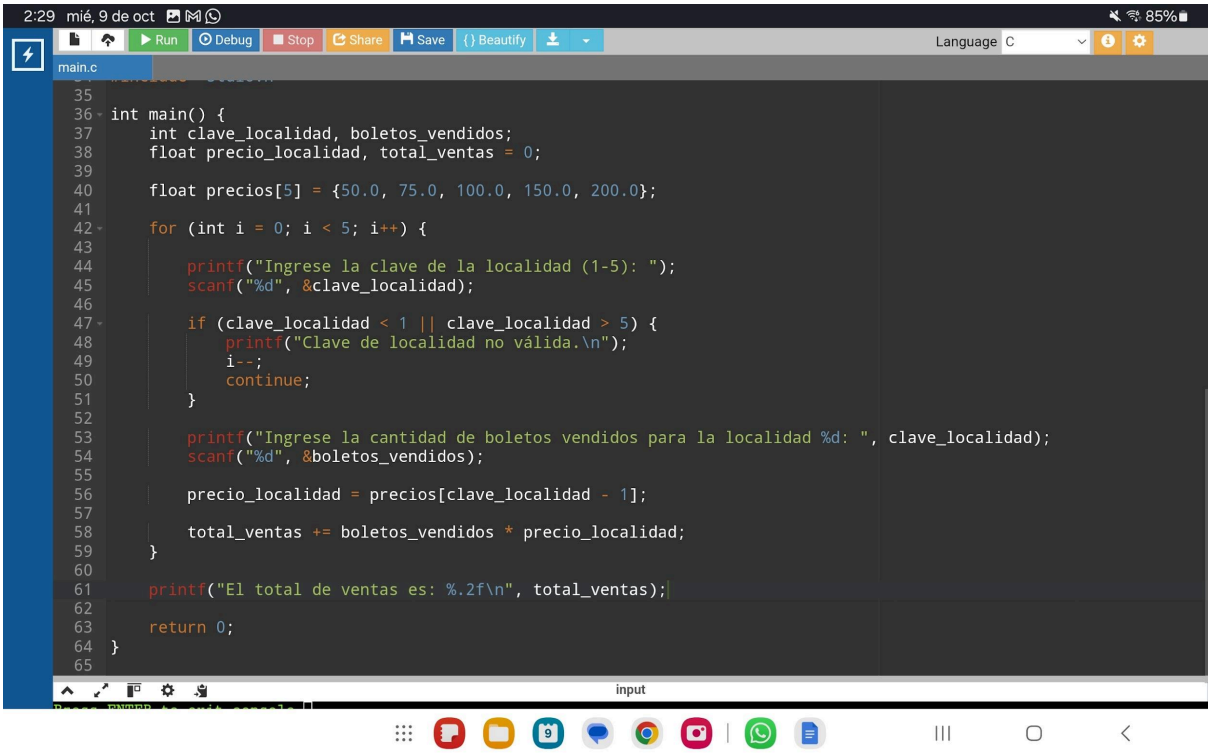
Ejercicio 1. Elabore un programa para calcular el término 10 de la secuencia de Fibonacci. Recuerde que los dos primeros números de la serie son 0 y 1. El resto se calcula con la suma de los dos números inmediatos que le preceden

```
/* INICIO
    Definir n como 10
    Crear un arreglo fib de tamaño n
    fib[0] ← 0
    fib[1] ← 1

    PARA i desde 2 HASTA n - 1 HACER
        fib[i] ← fib[i - 1] + fib[i - 2]
    FIN PARA

    IMPRIMIR "El término 10 de la secuencia de Fibonacci es:", fib[9]
FIN*/

 #include <stdio.h>

int main() {
    int n = 10;
    int fib[n];
    fib[0] = 0;
    fib[1] = 1;


    for (int i = 2; i < n; i++) {
        fib[i] = fib[i - 1] + fib[i - 2];
    }

    printf("El término 10 de la secuencia de Fibonacci es: %d\n", fib[9]);

    return 0;
}
```

Ejercicio 2. En un estadio se tienen 5 tipos diferentes de localidades, las cuales se identifican por una clave numérica que es un valor comprendido entre 1 y 5. Los precios de cada localidad y los datos referentes a las ventas de boletos para el próximo juego son:

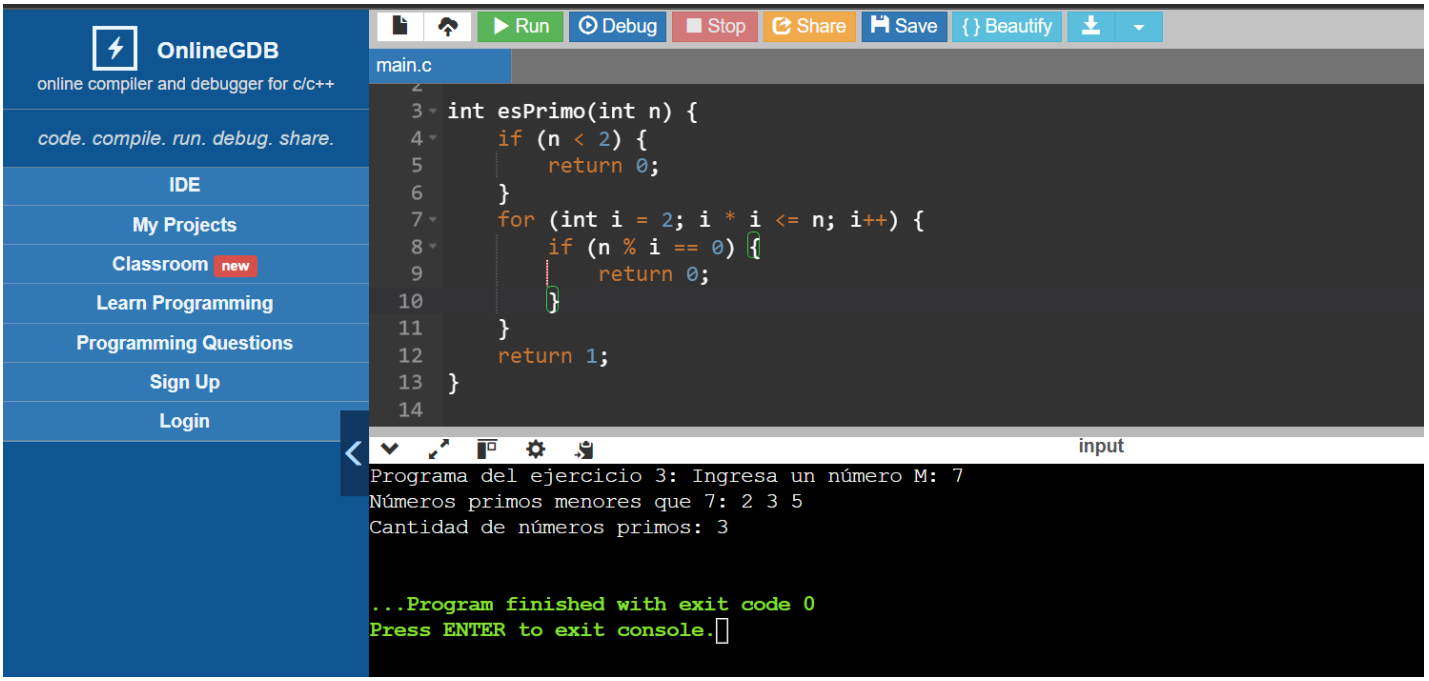

2:29 mié, 9 de oct
85%
Run
Debug
Stop
Share
Save
{} Beautify
Language C
main.c
/*Arteaga Gonzales Jaime Alejandro, GarcíaGarcía Jiménez Joel David, Pérez Alan y
Valencia Francisco Valentín
Ejercicio 2. En un estadio se tienen 5 tipos diferentes de localidades,
las cuales se identifican por una clave numérica que es un valor comprendido entre 1 y 5.
Los precios de cada localidad y los datos referentes a las ventas de boletos
para el próximo juego son:
Inicio
Definir clave de localidad, boletos vendidos como enteros
Definir precio de localidad, total ventas como reales
Asignar 0 a total ventas
Definir precios como arreglo de 5 reales [50.0, 75.0, 100.0, 150.0, 200.0]
Para i desde 0 hasta 4 hacer
Escribir "Ingrese la clave de la localidad (1-5): "
Leer clave de localidad
Si clave_localidad es menor que 1 o mayor que 5 entonces
Escribir "Clave de localidad no válida."
Restar 1 a i
Continuar
FinSi
Escribir "Ingrese la cantidad de boletos vendidos para la localidad ", clave de localidad, ": "
Leer boletos vendidos
Asignar precios[clave localidad - 1] a precio de localidad
Asignar total_ventas + (boletos vendidos * precio localidad) a total ventas
FinPara
input

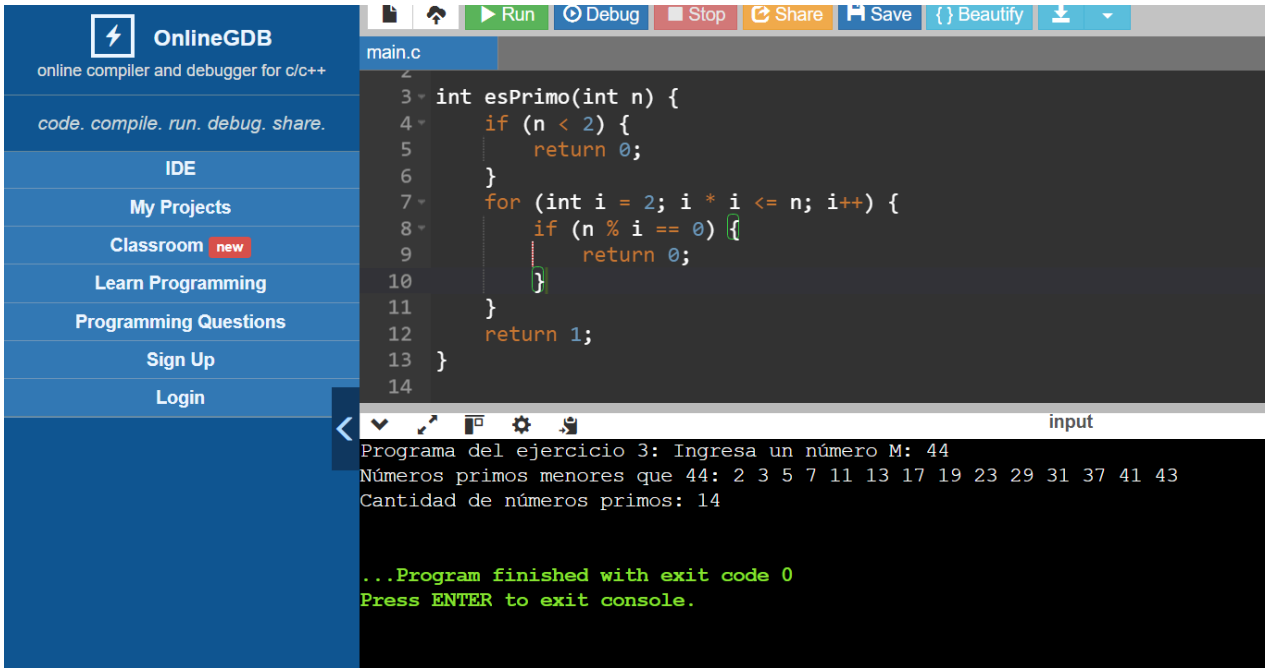

2:29 mié, 9 de oct
85%
Run
Debug
Stop
Share
Save
{} Beautify
Language C
main.c
Escribir "El total de ventas es: ", total_ventas
Fin*/
#include <stdio.h>
int main() {
int clave_localidad, boletos_vendidos;
float precio_localidad, total_ventas = 0;
float precios[5] = {50.0, 75.0, 100.0, 150.0, 200.0};
for (int i = 0; i < 5; i++) {
printf("Ingrese la clave de la localidad (1-5): ");
scanf("%d", &clave_localidad);
if (clave_localidad < 1 || clave_localidad > 5) {
printf("Clave de localidad no válida.\n");
i--;
continue;
}
printf("Ingrese la cantidad de boletos vendidos para la localidad %d: ", clave_localidad);
scanf("%d", &boletos_vendidos);
precio_localidad = precios[clave_localidad - 1];
total_ventas += boletos_vendidos * precio_localidad;
}
printf("El total de ventas es: %.2f\n", total_ventas);
input



https://onlinegdb.com/hr6TMDRCg

Ejercicio 3. Un número positivo N es un número primo si los únicos enteros positivos que lo dividen son exactamente 1 y N. Realice un programa que dado un número M, se obtenga y cuente todos los números primos menores a M.





Jaime: En este caso comenzamos hacer programas con estructuras de repetición, algo esencial en programación y algo que era más que claro que teníamos que llegar a eso, ya que como le hemos dicho anteriormente, cada dia y cada vez iremos avanzando más respecto a las estructuras, modos, conceptos y todo lo que implica a nuestra forma de programar en el lenguaje C. Las estructuras de repetición son algo que nos ayudará para poder hacer muchos programas, estas nos permiten repetir una acción (o grupo de acciones) varias veces. En cada uno de los problemas las estructuras de repetición nos ayudarán a repetir una acción como en el caso de los boletos o entros problemas que utilizamos acción como mientras, desde-hasta y repetir-hasta. Esta práctica nos muestra esta estructura que sin duda es de gran ayuda respecto a lo que tenemos que trabajar en problemas que requieran repetirse o hacer algo y poner un hasta donde, en general es interesante conocer más sobre estas estructuras y la práctica como brigada fue un éxito y los objetivos cumplidos de manera que como alumno sigamos aprendiendo más respecto a C.

Valentín: A lo largo de esta práctica, logramos entender la importancia de las estructuras de repetición en la programación en C. Con ejercicios como la secuencia de Fibonacci y la gestión de ventas de boletos en un estadio, pudimos ver cómo estas estructuras nos ayudan a resolver problemas repetitivos de manera eficiente. Además, aprendimos a usar diferentes tipos de bucles como *while*, *do-while* y *for*, lo que nos permitió observar sus

diferencias y cuándo es mejor usar cada uno. En resumen, fue una práctica que no solo nos ayudó a consolidar conceptos importantes, sino que también reforzó nuestra lógica para abordar problemas complejos.

Joel: Se cumplió el objetivo propuesto a través de los problemas planteados, los cuales permiten comprender de mejor manera el funcionamiento de las estructuras de repetición, su importancia en la resolución de problemas y su uso, así como su anidamiento en caso de ser necesario. La práctica constante facilita la identificación más rápida de la estructura adecuada para los distintos problemas que se presentan. Gracias a las indicaciones proporcionadas durante la práctica y a las clases previas del profesor, se logró alcanzar el objetivo inicial de manera satisfactoria, desarrollando un programa en lenguaje C que resolvió los problemas planteados. Además, contribuyó al fortalecimiento del pensamiento lógico-matemático y a un mejor conocimiento del lenguaje C.

Alan: Las estructuras de repetición en C, destaca la importancia de estas herramientas en la programación. Los integrantes del equipo reflexionan sobre cómo las estructuras como while, do-while y for permiten resolver problemas repetitivos de manera eficiente. Además, mencionan que la práctica constante y la comprensión de cuándo utilizar cada tipo de estructura son fundamentales para abordar problemas más complejos.
A través de ejercicios como la secuencia de Fibonacci y la gestión de boletos, el equipo reforzó su lógica y habilidad para programar en C, alcanzando los objetivos de la práctica.

BIBLIOGRAFÍAS:

O., C. (s. f.). *Sentencias Repetitivas*. UNIVERSIDAD VERACRUZANA.

https://lumen.uv.mx/recursoseducativos/EstructuraMientras/contenido.html#:~:text=Las%20estructuras%20repetitivas%2C%20permiten%20repetir,%2Dhasta%20y%20repetir%2Dhasta.