UNIVERSIDAD NACIONAL AUTÓNOMA DE MÉXICO

FACULTAD DE INGENIERÍA

Calculadora

Estructura de Datos y Algoritmos I

Grupo: 11

Práctica No. 1

Alumno:
López Morales Fernando Samuel

Profesor:
Gabriel Castillo Hernández

2024 - 2

Ciudad de México a 18 de febrero de 2024

ÍNDICE

PLANTEAMIENTO DEL PROBLEMA

Hacer un programa que opere 2 variables de tipo real en las 5 opciones planteadas:

1. Sume ambos números
2. Reste el primero menos el segundo
3. Multiplique ambos números
4. Incremente el primer número
5. Ingrese los valores respectivos para cada variable

Para cada opción se emplearán funciones con paso de parámetros por valor o por referencia para las 2 variables que se operan y para la variable que contiene el valor de su operación de manera que se practique su uso.
(Opción 0 termina el programa)

SOLUCIÓN

Se emplearán 3 archivos:

1. *principal.c* (Dado por el profesor y no podrá ser alterado más que para añadir bibliotecas.h)
2. *menu.c*:
    a. Contendrá la función `desplegarMenu()`: que desplegará las opciones en pantalla
    b. Contendrá la función `menu()`: que llamará a `desplegarMenu()` y pedirá el valor de la opción, mismo que devolverá para su manipulación en principal.c
3. *funciones.c*:
    a. Contiene las 5 funciones para cada una de las opciones descritas en PLANTEAMIENTO DEL PROBLEMA

- Se solicita al usuario el valor del caso al que quiere entrar, guardando dicho dato en la variable **`valorIngreso`**, para poder operar en base a él, corresponde a una opción 0-5 de las planteadas anteriormente.
- Se verifica que dicho valor ingresado por el usuario sea dentro del rango [0,5] y además que no sea un caracter o cadena.

```
desplegarMenu(1);
do{
    printf("Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer: \n");
    fflush(stdin); //No me funcionó
    n = scanf("%d", &valorIngreso);        //Guarda el valor de retorno de scanf para asegurar una entrada de datos exitosa
    if(n != 1 || (valorIngreso < 0 || valorIngreso > 5)){
        while((c = getchar()) != '\n' && c != EOF);
        printf("***Valor invalido, por favor ingresa solo alguna de las opciones del menu: (1-5)***\n");
        desplegarMenu(0);
    }
}while((valorIngreso < 0 || valorIngreso > 5)||n!=1);
```

- Se emplea la estructura condicional múltiple switch para poder decidir el caso de acuerdo al valor válido de **`valorIngreso.`**
- Se manipulan funciones con apuntadores para poder modificar directamente las variables originales presentes en principal.c

```c
float suma(float x, float y){
    printf("\n---Tus números han sido SUMADOS---\n");
    return x+y;
}
void resta(float * resultado, float x, float y){
    printf("\n---Tus números han sido RESTADOS---\n");
    *resultado = x-y;
}
void multiplicacion(float * primerFactor, float * segundoFactor, float * producto){
    printf("\n---Tus números han sido MULTIPLICADOS---\n");
    *producto = (*primerFactor)*(*segundoFactor);
}
void incremento(float * numero){
    printf("\n---Tu número ha sido INCREMENTADO---\n");
    *numero = *numero + 1;
}
```

- Se termina el programa solo si entra en el caso 5

ESTRUCTURAS O ALGORITMOS EMPLEADOS

Se emplearon tipos de datos simples y estructuras de control condicionales: `if`, `switch`, `do-while`.

CAPTURAS DEL PROGRAMA

Elección de opciones

Valor de **`valorIngreso`** incorrecto: 12 (fuera de rango)

```
===> BIENVENIDO A LA CALCULADORA INTELIGENTE DE LA FI <===
***Si ingresaste por primera vez recuerda actualizar los valores de X y Y con la opcion 5***

0.- SALIR Y TERMINAR PROGRAMA
1.- SUMA
2.- RESTA
3.- MULTIPLICACION
4.- INCREMENTO (sumarle 1)
5.- INGRESAR LOS VALORES DE X y Y
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
12
***Valor invalido, por favor ingresa solo alguna de las opciones del menu: (1-5)***

0.- SALIR Y TERMINAR PROGRAMA
1.- SUMA
2.- RESTA
3.- MULTIPLICACION
4.- INCREMENTO (sumarle 1)
5.- INGRESAR LOS VALORES DE X y Y
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
```

Valor de **`valorIngreso`** incorrecto: -2626.2 (fuera de rango)

```
0.- SALIR Y TERMINAR PROGRAMA
1.- SUMA
2.- RESTA
3.- MULTIPLICACION
4.- INCREMENTO (sumarle 1)
5.- INGRESAR LOS VALORES DE X y Y
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
-2626.2
***Valor invalido, por favor ingresa solo alguna de las opciones del menu: (1-5)***

0.- SALIR Y TERMINAR PROGRAMA
1.- SUMA
2.- RESTA
3.- MULTIPLICACION
4.- INCREMENTO (sumarle 1)
5.- INGRESAR LOS VALORES DE X y Y
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
```

Valor de **`valorIngreso`** incorrecto: “cinco” (cadena)

```
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
cinco
***Valor invalido, por favor ingresa solo alguna de las opciones del menu: (1-5)***

0.- SALIR Y TERMINAR PROGRAMA
1.- SUMA
2.- RESTA
3.- MULTIPLICACION
4.- INCREMENTO (sumarle 1)
5.- INGRESAR LOS VALORES DE X y Y
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
```

Elección de valores para X y Y (opción 5)

Valor de Y inválido por ser cadena:

```
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
5
Ingresa el valor de X: 40.2
Ingresa el valor de Y: cinco mil millones
Dato invalido, ingresa un numero real: Ingresa el valor de Y:
```

Elección correcta de ambos números

```
Dato invalido, ingresa un numero real: Ingresa el valor de Y: 5
=== Ahora X = 40.200001, y Y = 5.000000 ===
```

SUMA (Opción 1)

```
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
1

---Tus n├╢meros han sido SUMADOS---
40.200001 + 5.000000 = 45.200001

===> BIENVENIDO A LA CALCULADORA INTELIGENTE DE LA FI <===
```

RESTA (Opción 2)

```
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
2

---Tus n├╢meros han sido RESTADOS---
40.200001 - 5.000000 = 35.200001

===> BIENVENIDO A LA CALCULADORA INTELIGENTE DE LA FI <===
```

MULTIPLICACIÓN (Opción 3)

```
 ---Tus n├╢meros han sido MULTIPLICADOS---
 40.200001 * 5.000000 = 201.000000

 ===> BIENVENIDO A LA CALCULADORA INTELIGENTE DE LA FI <===
```

INCREMENTO (Opción 4)

```
Digita alguna de las anteriores opciones de acuerdo a lo que quieras hacer:
4

---Tu n├╢mero ha sido INCREMENTADO---
41.200001
```

ENTRADA
Números [0-5]

SALIDA
Resultado de la operación que se eligió

LIMITANTES (restricciones)

Ante la entrada de números con punto decimal cuando se requiere un entero, se toma el valor entero hasta el primer decimal encontrado.

Ejemplo
3.123.12, se interpreta como 3.

Ante la entrada de números con varios puntos cuando se requiere un número real, se toma el valor entero, y el número posterior al primer punto decimal hasta encontrar otro punto.

Ejemplo
2.42.3231.23, se interpreta com 2.42

López Morales Fernando Samuel *Facultad de Ingeniería*

REFLEXIÓN

La práctica tuvo como objetivo poder practicar el paso de parámetros por valor y por referencia, haciendo uso de los operadores de dirección **`(&)`** e indirección (**`*`**).
Resulta útil recordar las diferentes maneras de manejar valores en las funciones.
Para el ingreso de datos por la entrada estándar (teclado) se encuentra un problema si se intentan ingresar valores que no son los requeridos.

- Cuando se requiere un entero / real, si se ingresa un caracter, se abrirá un bucle infinito si no se tienen las precauciones

En mi caso, la manera de verificar el número de datos ingresados mediante el valor de retorno de la función **`scanf`** fue útil, sin embargo, por alguna razón que no logré explicar, **`fflush(stdin);`** no lograba limpiar el buffer, por lo que el ciclo infinito continuaba dados la entrada de caracteres.

Para asegurar que se vaciara el buffer, se construyó la línea:

```
while((c = getchar()) != '\n' && c != EOF);
```

a la cuál se ingresaría en caso de tener un valor 0 de retorno del **`scanf`**.

```
do{
    printf("Ingresa el valor de Y: ");
    fflush(stdin); //No me funciona
    datoCorrecto = scanf("%f", &segundoNumero);
    if(datoCorrecto!=1){
        while((c = getchar()) != '\n' && c != EOF);
        printf("Dato invalido, ingresa un numero real: ");
    }
}while(datoCorrecto!=1);
```

Lo que hará dicha línea es vaciar el buffer mediante la función **`getchar()`**, hasta que se encuentre el salto de línea o **`EOF`** que marca el final del archivo / entrada estándar.
Esto, a diferencia de **`fflush(stdin);`** sí lograba solucionar el problema del bucle infinito con una cadena de caracteres dada.

REFERENCIAS

- IBM documentation. (s. f.). scanf() https://www.ibm.com/docs/es/i/7.5?topic=functions-scanf-read-data
- IBM documentation. (s. f.). fflush() https://www.ibm.com/docs/es/i/7.5?topic=functions-fflush-write-buffer-file