LENGUAJE C

PROF. YULITH VANESSA ALTAMIRANO FLORES

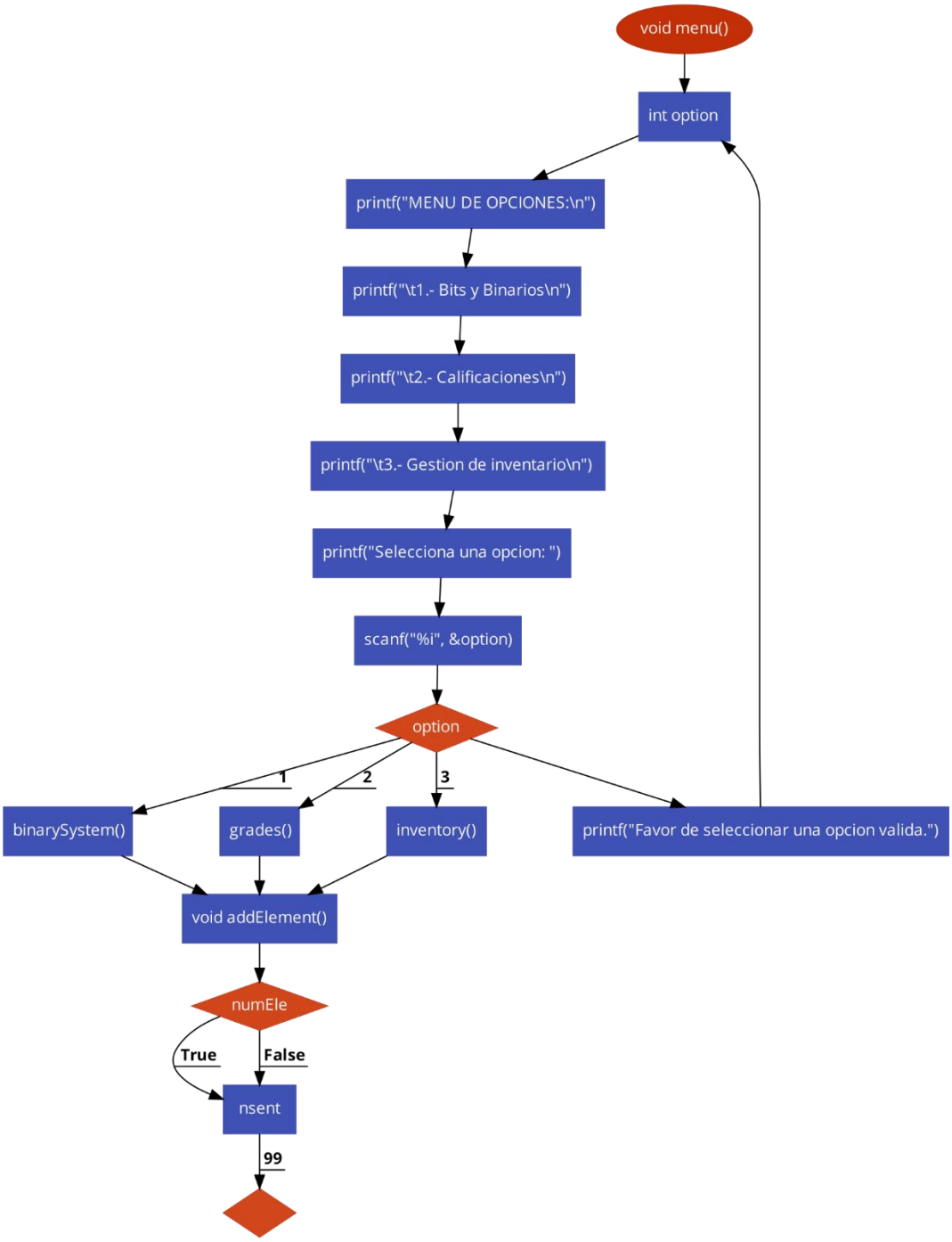
REPORTE DE PRACTICA
TIPOS, OPERADORES Y EXPRESIONES

KEVIN ALEJANDRO GONZALEZ TORRES
GRUPO 932

FUNCIÓN MENÚ()

{

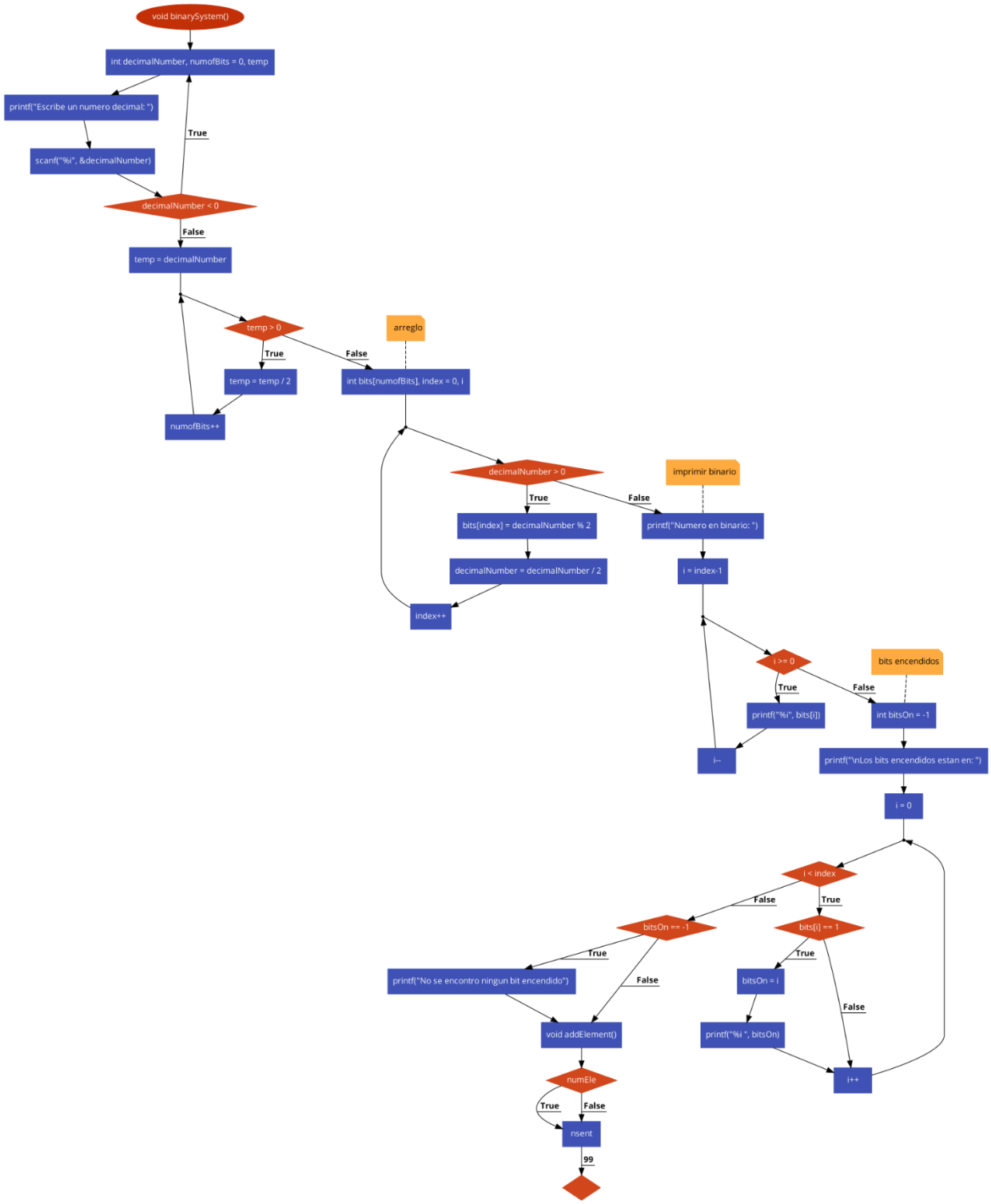
```c
void menu()
{
    start_menu:
    int option;

    printf("MENU DE OPCIONES:\n");
    printf("\t1.- Bits y Binarios\n");
    printf("\t2.- Calificaciones\n");
    printf("\t3.- Gestion de inventario\n");
    printf("Selecciona una opcion: ");
    scanf("%i", &option);

    switch (option)
    {
    case 1:
        binarySystem();
        break;

    case 2:
        grades();
        break;

    case 3:
        inventory();
        break;

    default:
        printf("Favor de seleccionar una opcion valida.");
        goto start_menu;
        break;
    }
}
```


void menu()
int option
printf("MENU DE OPCIONES:\n")
printf("\t1.- Bits y Binarios\n")
printf("\t2.- Calificaciones\n")
printf("\t3.- Gestion de inventario\n")
printf("Selecciona una opcion: ")
scanf("%i", &option)
option
1
2
3
binarySystem()
grades()
inventory()
printf("Favor de seleccionar una opcion valida.")
void addElement()
numEle
True
False
nsent
99

}

FUNCION NO_BINARIOS()
{

```c
void binarySystem()
{
    start:
    int decimalNumber, numofBits = 0, temp;

    printf("Escribe un numero decimal: ");
    scanf("%i", &decimalNumber);

    if (decimalNumber < 0)
    {
        goto start;
    }

    temp = decimalNumber;

    while (temp > 0)
    {
        temp = temp / 2;
        numofBits++;
    }

    // arreglo
    int bits[numofBits], index = 0, i;

    while (decimalNumber > 0)
    {
        bits[index] = decimalNumber % 2;
        decimalNumber = decimalNumber / 2;
        index++;
    }

    // imprimir binario
    printf("Numero en binario: ");

    for (i = index-1; i >= 0; i--)
    {
        printf("%i", bits[i]);
    }

    // bits encendidos
    int bitsOn = -1;

    printf("\nLos bits encendidos estan en: ");

    for (i = 0; i < index; i++)
    {
        if (bits[i] == 1)
        {
            bitsOn = i;
            printf("%i ", bitsOn);
        }
    }

    if (bitsOn == -1)
    {
        printf("No se encontro ningun bit encendido");
    }
}
```

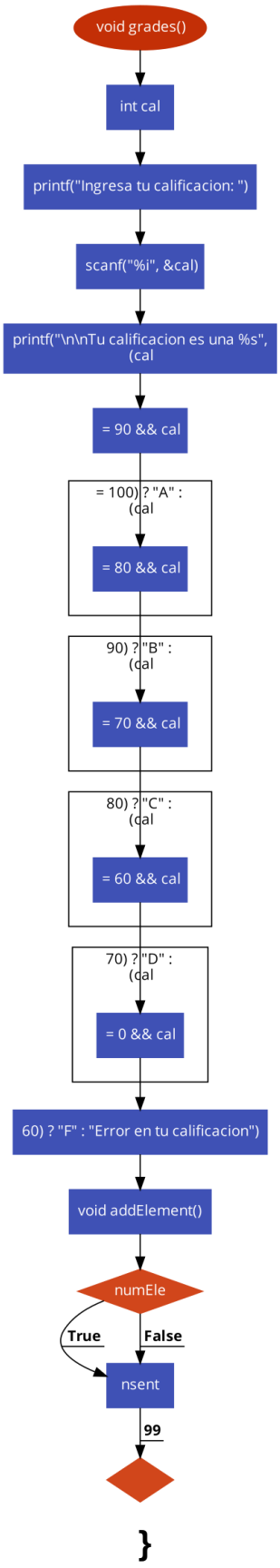

void binarySystem()
int decimalNumber, numofBits = 0, temp
printf("Escribe un numero decimal: ")
scanf("%i", &decimalNumber)
True
decimalNumber < 0
False
temp = decimalNumber
temp > 0
True
False
temp = temp / 2
numofBits++
arreglo
int bits[numofBits], index = 0, i
decimalNumber > 0
True
False
bits[index] = decimalNumber % 2
decimalNumber = decimalNumber / 2
index++
imprimir binario
printf("Numero en binario: ")
i = index-1
i >= 0
True
False
printf("%i", bits[i])
i--
bits encendidos
int bitsOn = -1
printf("\nLos bits encendidos estan en: ")
i = 0
i < index
False
True
bitsOn == -1
True
False
printf("No se encontro ningun bit encendido")
void addElement()
numEle
True
False
nsent
99
bits[i] == 1
True
False
bitsOn = i
printf("%i ", bitsOn)
i++

}

FUNCION CALIFICACIONES()

{

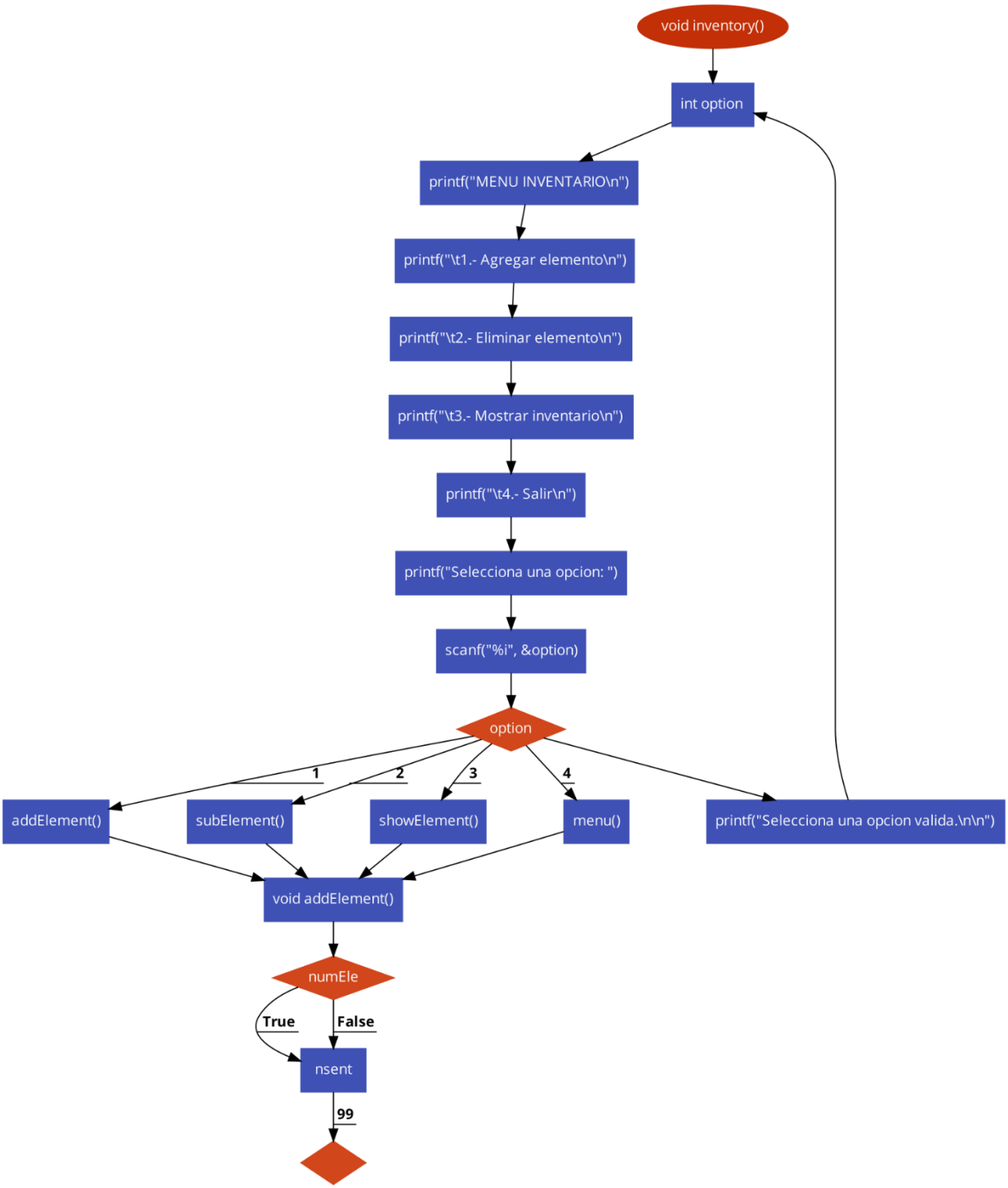
```c
void grades()
{
    int cal;
    printf("Ingresa tu calificacion: ");
    scanf("%i", &cal);
    printf("\n\nTu calificacion es una %s",
        (cal >= 90 && cal <= 100) ? "A" :
        (cal >= 80 && cal < 90) ? "B" :
        (cal >= 70 && cal < 80) ? "C" :
        (cal >= 60 && cal < 70) ? "D" :
        (cal >= 0 && cal < 60) ? "F" : "Error en tu calificacion");
}
```


void grades()
int cal
printf("Ingresa tu calificacion: ")
scanf("%i", &cal)
printf("\n\nTu calificacion es una %s",
(cal
= 90 && cal
= 100) ? "A" :
(cal
= 80 && cal
90) ? "B" :
(cal
= 70 && cal
80) ? "C" :
(cal
= 60 && cal
70) ? "D" :
(cal
= 0 && cal
60) ? "F" : "Error en tu calificacion")
void addElement()
numEle
True
False
nsent
99

}

FUNCION INVENTARIO_1()
{

```c
void inventory()
{
    start_inv:
    int option;

    printf("MENU INVENTARIO\n");
    printf("\t1.- Agregar elemento\n");
    printf("\t2.- Eliminar elemento\n");
    printf("\t3.- Mostrar inventario\n");
    printf("\t4.- Salir\n");

    printf("Selecciona una opcion: ");
    scanf("%i", &option);

    switch (option)
    {
    case 1:
        addElement();
        break;

    case 2:
        subElement();
        break;

    case 3:
        showElement();
        break;

    case 4:
        menu();
        break;

    default:
        printf("Selecciona una opcion valida.\n\n");
        goto start_inv;
        break;
    }
}
```

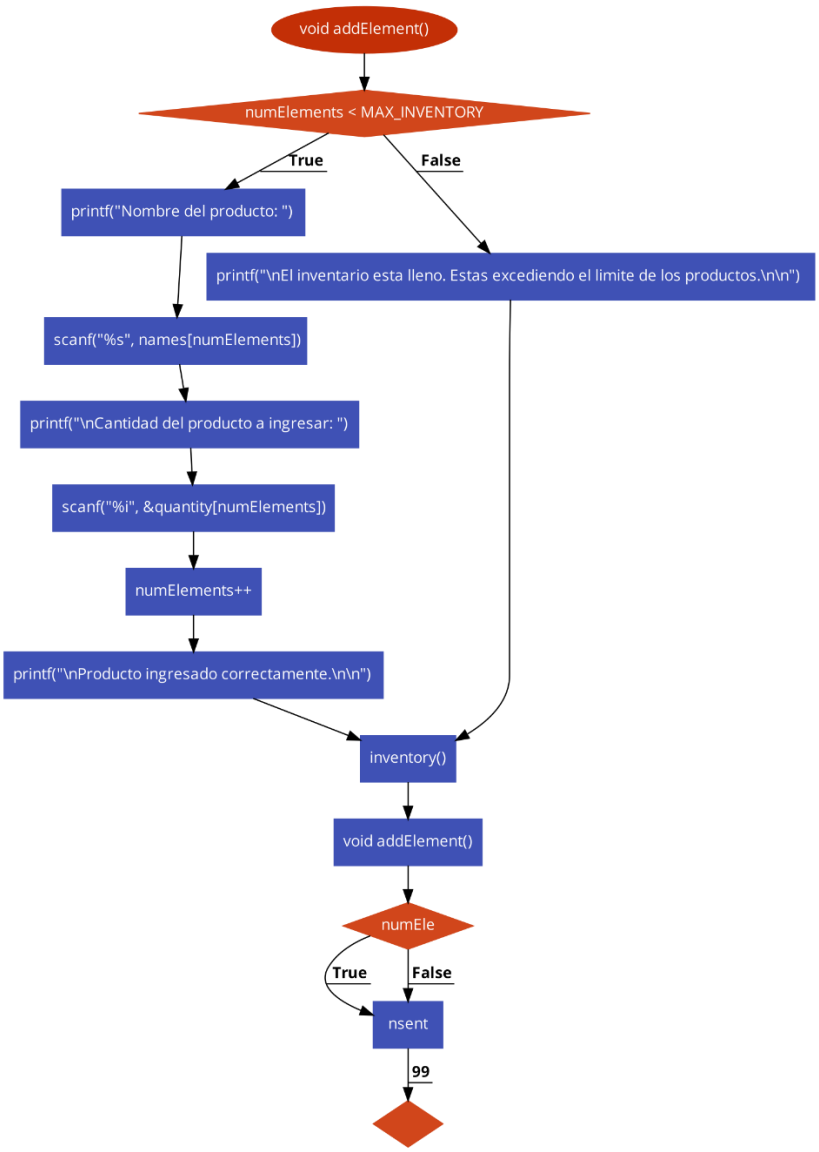

void inventory()
int option
printf("MENU INVENTARIO\n")
printf("\t1.- Agregar elemento\n")
printf("\t2.- Eliminar elemento\n")
printf("\t3.- Mostrar inventario\n")
printf("\t4.- Salir\n")
printf("Selecciona una opcion: ")
scanf("%i", &option)
option
1
2
3
4
addElement()
subElement()
showElement()
menu()
printf("Selecciona una opcion valida.\n\n")
void addElement()
numEle
True
False
nsent
99

}

FUNCION INVENTARIO_2()

{

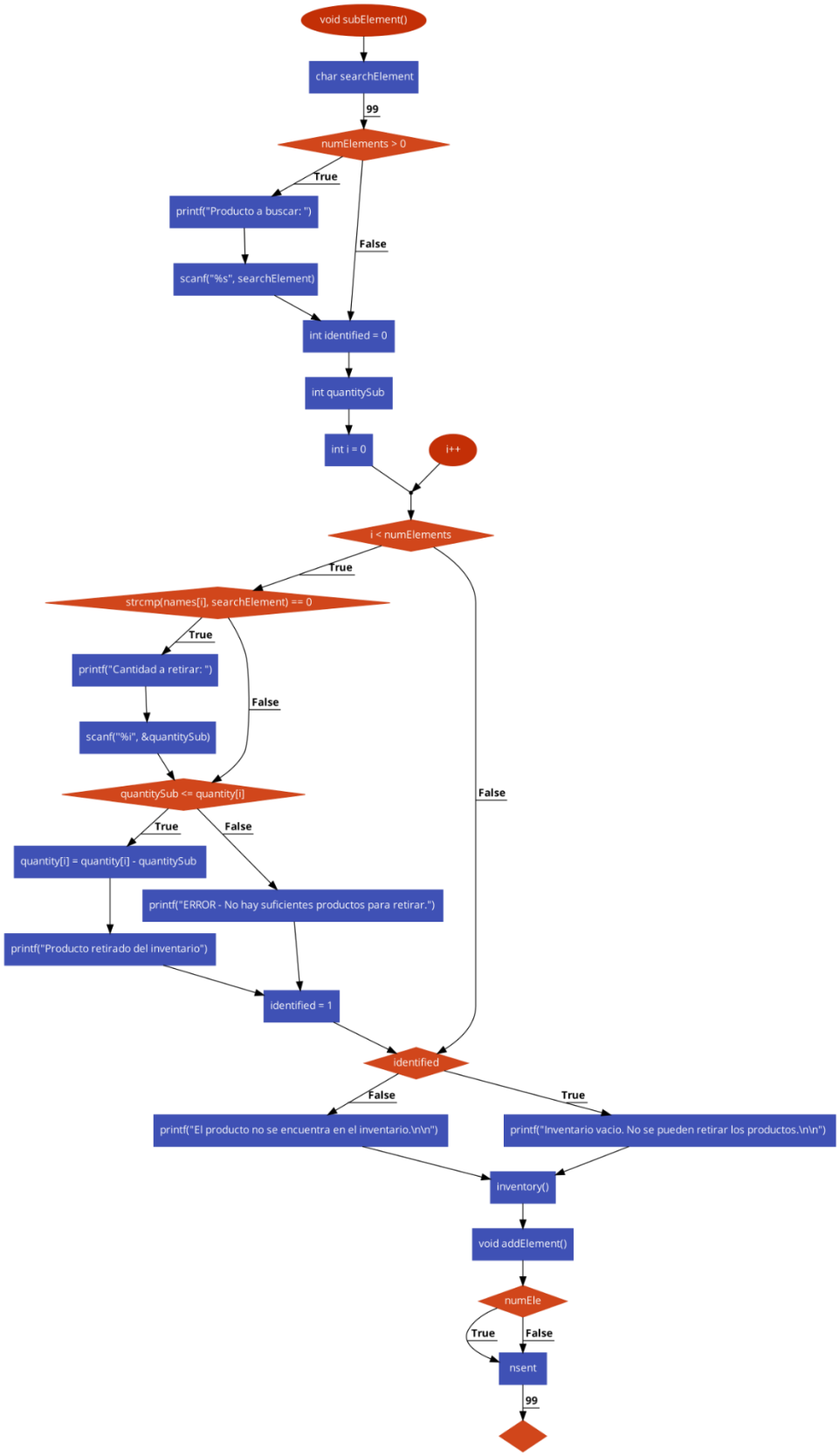
```c
void addElement()
{
    if (numElements < MAX_INVENTORY)
    {
        printf("Nombre del producto: ");
        scanf("%s", names[numElements]);
        printf("\nCantidad del producto a ingresar: ");
        scanf("%i", &quantity[numElements]);
        numElements++;
        printf("\nProducto ingresado correctamente.\n\n");
    }
    else
    {
        printf("\nEl inventario esta lleno. Estas excediendo el limite de los productos.\n\n");
    }
    inventory();
}
```



}

FUNCION INVENTARIO_3()
{

```c
void subElement()
{
    char searchElement[99];

    if (numElements > 0)
    {
        printf("Producto a buscar: ");
        scanf("%s", searchElement);
    }

    int identified = 0;
    int quantitySub;

    for (int i = 0; i < numElements; i++)
    {
        if (strcmp(names[i], searchElement) == 0)
        {
            printf("Cantidad a retirar: ");
            scanf("%i", &quantitySub);
        }

        if (quantitySub <= quantity[i])
        {
            quantity[i] = quantity[i] - quantitySub;
            printf("Producto retirado del inventario");
            identified = 1;
            break;
        }
        else
        {
            printf("ERROR - No hay suficientes productos para retirar.");
            identified = 1;
            break;
        }
    }

    if (!identified)
    {
        printf("El producto no se encuentra en el inventario.\n\n");
    }
    else
    {
        printf("Inventario vacio. No se pueden retirar los productos.\n\n");
    }
    inventory();
}
```

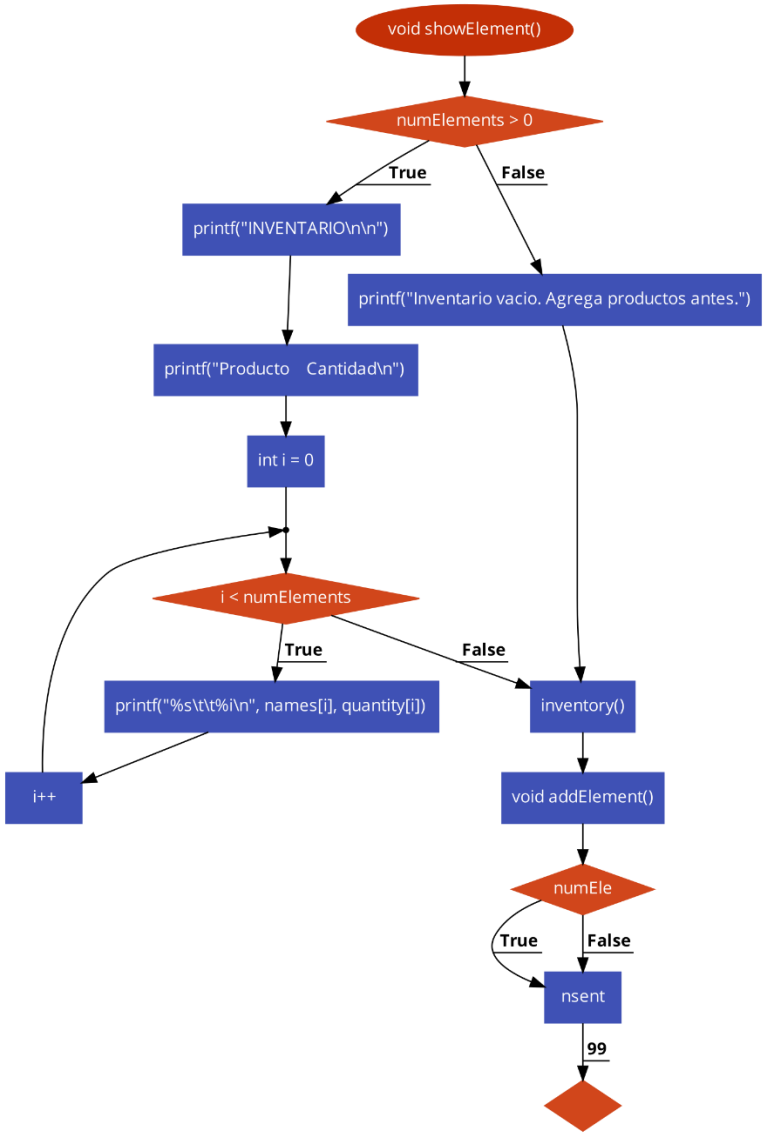

void subElement()
char searchElement
99
numElements > 0
True
False
printf("Producto a buscar: ")
scanf("%s", searchElement)
int identified = 0
int quantitySub
int i = 0
i++
i < numElements
True
False
strcmp(names[i], searchElement) == 0
True
False
printf("Cantidad a retirar: ")
scanf("%i", &quantitySub)
quantitySub <= quantity[i]
True
False
quantity[i] = quantity[i] - quantitySub
printf("ERROR - No hay suficientes productos para retirar.")
printf("Producto retirado del inventario")
identified = 1
identified
False
True
printf("El producto no se encuentra en el inventario.\n\n")
printf("Inventario vacio. No se pueden retirar los productos.\n\n")
inventory()
void addElement()
numEle
True
False
nsent
99

}

FUNCION INVENTARIO_4()

{

```c
void showElement()
{
    if(numElements > 0)
    {
        printf("INVENTARIO\n\n");
        printf("Producto    Cantidad\n");
        for (int i = 0; i < numElements; i++)
        {
            printf("%s\t\t%i\n", names[i], quantity[i]);
        }
        inventory();
    }
    else
    {
        printf("Inventario vacio. Agrega productos antes.");
        inventory();
    }
}
```



}