



PROGRAMACION
TDSD214

ASIGNATURA:	Programación
PROFESOR:	Ing. Ivonne Maldonado
PERÍODO ACADÉMICO:	2023-B

LABORATORIO - 7

TÍTULO:
FUNCIONES

```
.....  
if (bisiesto(anio)) //llamada a la función  
    cout << "Bisiesto" << endl;  
else  
    cout << "No es bisiesto" << endl;  
    system("pause");  
}  
int bisiesto(int a) //definición de la función  
{  
    if (a%4==0 and a%100!=0 or a%400==0)  
        return 1;  
    else  
        return 0;  
}
```

Nombre:

Guerra Lovato Josué

PROPÓSITO DE LA PRÁCTICA

Familiarizar al estudiante con el uso de funciones en el lenguaje C++.

OBJETIVO GENERAL

Utilizar funciones en la solución de problemas.

INSTRUCCIONES

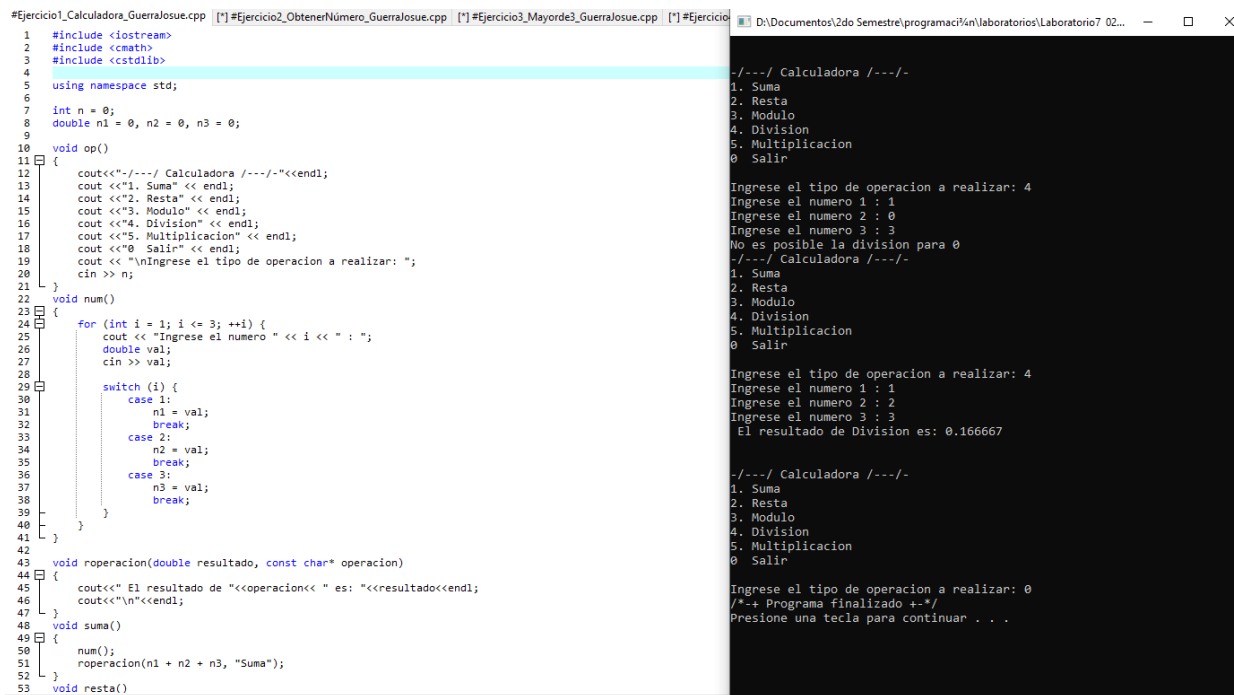
- Declarar funciones
- Definir funciones
- Llamar funciones

TAREA

Haciendo uso de funciones, resuelva cada uno de los siguientes ejercicios:

- Calculadora básica (al menos 5 funciones).

Dentro de dicho programa se definen funciones las cuales se ejecutan en el int main solo declarando las funciones.



```
#Ejercicio1_Calculadora_Guerrajose.cpp | #Ejercicio2_ObtenerNúmero_Guerrajose.cpp | #Ejercicio3_Mayorde3_Guerrajose.cpp | #Ejercicio4_Mayorde3_Guerrajose.cpp | #Ejercicio5_Mayorde3_Guerrajose.cpp
1  #include <iostream>
2  #include <cmath>
3  #include <cstdlib>
4
5  using namespace std;
6
7  int n = 0;
8  double n1 = 0, n2 = 0, n3 = 0;
9
10 void op()
11 {
12     cout<<"---/ Calculadora /---/"<<endl;
13     cout<<"1. Suma" << endl;
14     cout<<"2. Resta" << endl;
15     cout<<"3. Modulo" << endl;
16     cout<<"4. Division" << endl;
17     cout<<"5. Multiplicacion" << endl;
18     cout<<"0 Salir" << endl;
19     cout<<"\nIngrese el tipo de operacion a realizar: ";
20     cin >> n;
21 }
22
23 void num()
24 {
25     for (int i = 1; i <= 3; ++i) {
26         cout<<"Ingrese el numero " << i << " : ";
27         double val;
28         cin >> val;
29
30         switch (i) {
31             case 1:
32                 n1 = val;
33                 break;
34             case 2:
35                 n2 = val;
36                 break;
37             case 3:
38                 n3 = val;
39                 break;
40         }
41     }
42 }
43
44 void roperacion(double resultado, const char* operacion)
45 {
46     cout<<" El resultado de "<<operacion<<" es: "<<resultado<<endl;
47     cout<<"\n"<<endl;
48 }
49
50 void suma()
51 {
52     num();
53     roperacion(n1 + n2 + n3, "Suma");
54 }
55
56 void resta()
57 {
58     num();
59     roperacion(n1 - n2 - n3, "Resta");
60 }
61
62 void modulo()
63 {
64     num();
65     roperacion(n1 % n2 % n3, "Modulo");
66 }
67
68 void division()
69 {
70     num();
71     roperacion(n1 / n2 / n3, "Division");
72 }
73
74 void multiplicacion()
75 {
76     num();
77     roperacion(n1 * n2 * n3, "Multiplicacion");
78 }
79
80 void salir()
81 {
82     cout<<"\nPrograma finalizado +*/*\n";
83     cout<<"Presione una tecla para continuar . . .";
84     cin.get();
85 }
86
87 int main()
88 {
89     op();
90     num();
91     suma();
92     resta();
93     modulo();
94     division();
95     multiplicacion();
96     salir();
97 }
```

```
---/ Calculadora /---/
1. Suma
2. Resta
3. Modulo
4. Division
5. Multiplicacion
0 Salir
Ingrese el tipo de operacion a realizar: 4
Ingrese el numero 1 : 1
Ingrese el numero 2 : 0
Ingrese el numero 3 : 3
No es posible la division para 0
---/ Calculadora /---/
1. Suma
2. Resta
3. Modulo
4. Division
5. Multiplicacion
0 Salir
Ingrese el tipo de operacion a realizar: 4
Ingrese el numero 1 : 1
Ingrese el numero 2 : 2
Ingrese el numero 3 : 3
El resultado de Division es: 0.166667
---/ Calculadora /---/
1. Suma
2. Resta
3. Modulo
4. Division
5. Multiplicacion
0 Salir
Ingrese el tipo de operacion a realizar: 0
/*+ Programa finalizado +*/*
Presione una tecla para continuar . . .
```

```

53 void resta()
54 {
55     num();
56     roperacion(n1 - n2 - n3, "Resta");
57 }
58 void modulo()
59 {
60     cout<<"El ejercicio solo toma""\n\los 2 primeros para caclicular"<<endl;
61     num();
62     roperacion(fmod(n1, n2), "Modulo");
63 }
64 void division()
65 {
66     do{
67         num();
68         if (n2 == 0)
69         {
70             cout<<"No es posible la division para 0"<<endl;
71         }else{
72             roperacion((n1 / n2) / n3 , "Division");
73         }
74     }while (n3 == 0);
75 }
76 void multiplicacion()
77 {
78     num();
79     roperacion(n1 * n2 * n3, "Multipliacion");
80 }
81
82 int main ()
83 {
84     do{
85         op();
86         switch (n){
87             case 0:
88                 cout<<"/*-+ Programa finalizado +-*/"<<endl;
89                 system("pause");
90                 break;
91             case 1:
92                 suma();
93                 break;
94             case 2:
95                 resta();
96                 break;
97             case 3:
98                 modulo();
99                 break;
100             case 4:
101                 division();
102                 break;
103             case 5:
104                 multiplicacion();
105                 break;
106             default:
107                 cout<<"\nOpcion no encontrada\n"<<endl;
108                 cout<<"Ingrese nuevmanete cualquiera de las opciones"<<endl;
109                 break;
110         }
111     }while (n != 0);
112     return 0;
113 }

```

- Ingresado un numero decimal, que devuelva en partes el número, por ejemplo:

123,567	500 decimas
100 centenas	60 centésimas
20 decenas	7 milésimas
3 unidades	

El programa como tal cumple con lo solicitado mostrando en pantalla con cada uno de los dígitos a que parte corresponde, dentro del int main solo se ubica el texto donde mostrara al usuario como debe ser.

LAS IMÁGENES SE ENCUENTRAN EN LA SIGUIENTE PÁGINA.

```
#Ejercicio1_Calculadora_Guerralosue.cpp #Ejercicio2_ObtenerNúmero_Guerralosue.cpp #Ejercicio3_Mayorde3_Guerralosue.cpp #Ejercicio4_CartaIntermedia_Guerralosue.cpp
1 #include <iostream>
2 #include <cmath>
3
4 using namespace std;
5
6 int ParteEntera(double numero) {
7     return static_cast<int>(numero);
8 }
9 int Centenas(double numero) {
10    return abs(numero) / 100;
11 }
12 int Decenas(double numero) {
13    return static_cast<int>(abs(numero) / 10) % 10;
14 }
15 int Unidades(double numero) {
16    return static_cast<int>(abs(numero)) % 10;
17 }
18 double ParteDecimal(double numero) {
19    return abs(numero - static_cast<int>(numero));
20 }
21 int Decimas(double numero) {
22    return abs(ParteDecimal(numero) * 1000) / 100;
23 }
24 int Centesimas(double numero) {
25    return static_cast<int>(abs(static_cast<int>(ParteDecimal(numero) * 1000) / 10) % 10);
26 }
27
28 int Milesimas(double numero) {
29    return static_cast<int>(abs(static_cast<int>(ParteDecimal(numero) * 1000) % 10));
30 }
31
32 int main() {
33    -- Bienvenido --
34    cout<<"El programa indicara el numero por partes"<<endl;
35    double numero_a_separar;
36    cout<<"Ingrese un numero: ";
37    cin>>numero_a_separar;
38    cout<<"\n";
39
40    cout<<"Parte entera: " << ParteEntera(numero_a_separar) << endl;
41    cout<<"Centenas(numero_a_separar) << (Centenas(numero_a_separar) == 1 ? " Centena" : " Centenas") << endl;
42    cout<<"Decenas(numero_a_separar) << (Decenas(numero_a_separar) == 1 ? " Decena" : " Decenas") << endl;
43    cout<<"Unidades(numero_a_separar) << (Unidades(numero_a_separar) == 1 ? " Unidad" : " Unidades") << endl;
44    cout<<"\n";
45
46    cout<<"Parte decimal: " << ParteDecimal(numero_a_separar) << endl;
47    cout<<"Decimas(numero_a_separar) << (Decimas(numero_a_separar) == 1 ? " Decima" : " Decimas") << endl;
48    cout<<"Centesimas(numero_a_separar) << (Centesimas(numero_a_separar) == 1 ? " Centesima" : " Centesimas") << endl;
49    cout<<"Milesimas(numero_a_separar) << (Milesimas(numero_a_separar) == 1 ? " Milesima" : " Milesimas") << endl;
50
51    return 0;
52 }
```

```
-- Bienvenido --
El programa indicara el numero por partes
Ingrese un numero: 1648.28

Parte entera: 1648
16 Centenas
4 Decenas
8 Unidades

Parte decimal: 0.28
2 Decimas
7 Centesimas
0 Milesimas

-----
Process exited with return value 0
Press any key to continue . . .
```

o Mayor de 3 números.

```
#Ejercicio1_Calculadora_Guerralosue.cpp #Ejercicio2_ObtenerNúmero_Guerralosue.cpp #Ejercicio3_Mayorde3_Guerralosue.cpp
1 #include <iostream>
2 using namespace std;
3
4 int mayor(int x, int y, int z) {
5     if (x > y && x > z) {
6         return x;
7     } else if (y > x && y > z) {
8         return y;
9     } else {
10        return z;
11    }
12 }
13 int medio(int x, int y, int z) {
14    if ((x > y && x < z) || (x > z && x < y)) {
15        return x;
16    } else if ((y > x && y < z) || (y > z && y < x)) {
17        return y;
18    } else {
19        return z;
20    }
21 }
22 int menor(int x, int y, int z) {
23    if (x < y && x < z) {
24        return x;
25    } else if (y < x && y < z) {
26        return y;
27    } else {
28        return z;
29    }
30 }
31 void num(int &n1, int &n2, int &n3)
32 {
33    for (int i = 1; i <= 3; ++i) {
34        cout<<"Ingrese el numero " << i <<" : ";
35        double val;
36        cin>>val;
37        switch (i) {
38            case 1:
39                n1 = val;
40                break;
41            case 2:
42                n2 = val;
43                break;
44            case 3:
45                n3 = val;
46                break;
47        }
48    }
49 }
50
51 int main() {
52    int a = 0, b = 0, c = 0;
53
54    cout<<" -- Bienvenido -- " <<endl;
55    cout<<"El programa indicara el numero mayor de 3"<<endl;
56    cout<<"\n";
57    num(a, b, c);
58    cout<<"\n";
59    cout<<"El mayor de los numeros " << a <<" | " << b <<" | " << c <<" es: " << mayor(a, b, c) << endl;
60    cout<<"Con el siguiente orden de mayor a menor:" << endl;
61    cout<< mayor(a, b, c) <<" > " << medio(a, b, c) <<" > " << menor(a, b, c) <<" ";
62
63    return 0;
64 }
```

```
-- Bienvenido --
El programa indicara el numero mayor de 3

Ingrese el numero 1 : 16
Ingrese el numero 2 : 24
Ingrese el numero 3 : 12

El mayor de los numeros 16 | 24 | 12 es: 24
Con el siguiente orden de mayor a menor:
24 > 16 > 12

-----
Process exited with return value 0
Press any key to continue . . .
```

- Carta intermedia.
- El juego inicia con el ingreso del monto total a jugar (monto total a gastar como máximo en el juego).
- Antes de cada lanzamiento el usuario deberá ingresar el valor a “apostar”. Recuerde no puede apostar en un lanzamiento más del 30% de su dinero en bolsa.
- Con la puesta en la mesa, el computador mostrará 2 “cartas” al azar, y el usuario deberá decir si juega o no.
- ✦ Si el usuario decide jugar, la computadora apostará el doble que apostó el jugador.
- ✦ Si el usuario decide “pasar”, el 5% de su apuesta se queda en bolsa y el resto es devuelto para la siguiente ronda.
- El usuario gana cuando la siguiente carta “mostrada” por el computador es una “carta intermedia” (valor entre las otras dos cartas).
- La casa gana cuando la siguiente carta “mostrada” por el computador NO es una “carta intermedia” (valor entre las otras dos cartas).
- El usuario pierde cuando en su “bolsillo” tiene menos del 3% del monto total a jugar o cuando ha perdido 3 veces seguidas.

```
#Ejercicio1_Calculadora_GuerraJosue.cpp #Ejercicio2_ObtenerNúmero_GuerraJosue.cpp [*] #Ejercicio3_Mayorde3_GuerraJosue.cpp [*] #Ejercicio4_CartaIntermedia_GuerraJosue.cpp
1  #include <iostream>
2  #include <cstdlib>
3  #include <ctime>
4  #include <set>
5  #include <windows.h>
6
7  using namespace std;
8
9  double m = 0, v = 0;
10 int numPerdidas = 0;
11 string nombre;
12
13 void inicio()
14 {
15     cout<<"    ---// Bienvenido //--    "<<endl;
16     cout<<" Juego de la carta intermedia"<<endl;
17     cout<<" Inserte su nombre para comenzar: "<<endl;
18     cin>>nombre;
19 }
20 void pedirMonto()
21 {
22     cout << "Ingrese el monto total a jugar: "<<endl;
23     cin >> m;
24 }
25
26 void valorApostar()
27 {
28     while (true)
29     {
30         cout << "Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): ";
31         cin >> v;
32         if (v <= m * 0.3)
33             break;
34         else
35             cout << "La cantidad ingresada no puede sobrepasar el 30% de su monto actual." << endl;
36     }
37 }
38
39 int cartaAleatoria(set<int> &cartasUsadas)
40 {
41     int carta;
42     do
43     {
44         carta = rand() % 10 + 1;
45     } while (cartasUsadas.count(carta) > 0);
46     cartasUsadas.insert(carta);
47     return carta;
48 }
49
50 bool cartaIntermedia(int a, int b, int c)
51 {
52     return (a < c && c < b) || (b < c && c < a);
53 }
```

LA CONTINUACIÓN DEL CÓDIGO SE ENCUENTRA EN LA SIGUIENTE HOJA

```

#Ejercicio1_Calculadora_GuerraJosue.cpp #Ejercicio2_ObtenerNúmero_GuerraJosue.cpp #Ejercicio3_Mayorde3_GuerraJosue.cpp [*] #Ejercicio4_CartaIntermedia_GuerraJosue.cpp
55
56 void mostrarApuestas(double apuestaJugador, double apuestaComputadora)
57 {
58     cout << "Apuesta jugador: " << apuestaJugador << endl;
59     cout << "Apuesta computadora: " << apuestaComputadora << endl;
60     cout<<"\n";
61 }
62
63 void mostrarResultado(double bolsa, double monto)
64 {
65     cout << "Bolsa: " << bolsa << " Monto restante: " << monto << endl;
66     cout<<"\n";
67 }
68
69 int main()
70 {
71     srand(time(0));
72     double bolsa = 0;
73
74     inicio();
75     pedirMonto();
76
77     do
78     {
79         int x, y, z;
80         set<int> cartasUsadas;
81
82         valorApostar();
83         m -= v;
84
85         x = cartaAleatoria(cartasUsadas);
86         y = cartaAleatoria(cartasUsadas);
87         z = cartaAleatoria(cartasUsadas);
88
89         cout << "Cartas en la mesa: " << x << " " << y << endl;
90
91         if (v > 0)
92         {
93             cout << "Desea seguir jugando? (Si/No): "<<endl;
94             string respuesta;
95             cin >> respuesta;
96             cout<<"\n";
97
98             if (respuesta == "No" || respuesta == "no")
99             {
100                 mostrarApuestas(v * 0.95, 0);
101                 bolsa += v * 0.05;
102                 v -= v * 0.05;
103             }
104             else
105             {
106                 mostrarApuestas(v, v * 2);
107             }
108         }
109
110         cout << "Carta siguiente: " << z << endl;
111         Sleep(3000);
112
113         if (cartaIntermedia(x, y, z))
114         {
115             cout << "¡Has ganado!" << endl;
116             bolsa += v * 2;
117             numPerdidas = 0;
118         }
119         else
120         {
121             cout << "La casa gana." << endl;
122             numPerdidas++;
123
124             if (m < m * 0.03 || numPerdidas >= 3)
125             {
126                 cout <<" Has perdido el juego "<<nombre<< endl;
127                 break;
128             }
129         }
130
131         mostrarResultado(bolsa, m);
132     } while (true);
133
134     return 0;
135 }
136

```

EJECUCIÓN DEL CÓDIGO EN LA SIGUIENTE HOJA

```

---// Bienvenido ---
Juego de la carta intermedia
Inserte su nombre para comenzar:
Josue
Ingrese el monto total a jugar:
1.00
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0.10
Cartas en la mesa: 6 8
Desea seguir jugando? (Si/No):
s
Apuesta jugador: 0.1
Apuesta computadora: 0.2
Carta siguiente: 1
La casa gana.
Bolsa: 0 Monto restante: 0.9
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0.25
Cartas en la mesa: 1 10
Desea seguir jugando? (Si/No):
n
Apuesta jugador: 0.25
Apuesta computadora: 0.5
Carta siguiente: 5
¡Has ganado!
Bolsa: 0.5 Monto restante: 0.65
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0.25
La cantidad ingresada no puede sobrepasar el 30% de su monto actual.
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0.15
Cartas en la mesa: 1 6
Desea seguir jugando? (Si/No):
n
Apuesta jugador: 0.15
Apuesta computadora: 0.3
Carta siguiente: 7
La casa gana.
Bolsa: 0.5 Monto restante: 0.5
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0.01
Cartas en la mesa: 4 10
Desea seguir jugando? (Si/No):
s
Apuesta jugador: 0.01
Apuesta computadora: 0.02
Carta siguiente: 8
¡Has ganado!
Bolsa: 1.1 Monto restante: 0.05
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0.05
La cantidad ingresada no puede sobrepasar el 30% de su monto actual.
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0.01
Cartas en la mesa: 7 4
Desea seguir jugando? (Si/No):
s
Apuesta jugador: 0.01
Apuesta computadora: 0.02
Carta siguiente: 8
La casa gana.
Bolsa: 1.1 Monto restante: 0.04
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0.01
Cartas en la mesa: 7 1
Desea seguir jugando? (Si/No):
s
Apuesta jugador: 0.01
Apuesta computadora: 0.02
Carta siguiente: 9
La casa gana.
Bolsa: 1.1 Monto restante: 0.03
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0.01
La cantidad ingresada no puede sobrepasar el 30% de su monto actual.
Ingrese el valor a apostar (menos del 30% de su dinero en bolsa): 0
Cartas en la mesa: 4 9
Carta siguiente: 3
La casa gana.
Has perdido el juego Josue
-----
Process exited with return value 0
Press any key to continue . . .

```

Cabe mencionar que el código se lo realiza con funciones tanto para pedir el nombre como para poder continuar con el juego, de dicho modo, también en la parte de la ejecución del programa se ponen capturas diferentes ya que el usuario iba ganando, y lo que se quería mostrar era la ejecución completa, para poder realizar este programa se observó un video el cual iba dando alguna idea de cómo poder realizarlo y del mismo modo también se estudio nuevamente lo expuesto por la ingeniera en clase [1], [2], [3]

CONCLUSIONES:

En resumen, el uso de funciones C para resolver problemas proporciona importantes beneficios en términos de modularización, reutilización de código y claridad estructural. Dividir un programa en funciones más pequeñas y específicas hace que sea más fácil comprender el código, identificar y corregir errores y reutilizar funciones en diferentes partes del programa o incluso en otros proyectos. Además, el uso de funciones promueve un diseño más manejable y mantenible, permitiendo a los programadores resolver problemas de una manera más eficiente y estructurada. El código agregado crea funciones como `mayor`, `medio`, `menor`, `inicio`, `pedirMonto`, `valorApostar`, `cartaAleatoria`, `cartaIntermedia`, `mostrarApuestas`, `mostrarResultado`, que demuestran cómo las funciones de modularización pueden mejorar la legibilidad y organización de los programas en C++

RECOMENDACIONES:

Dados los beneficios obvios de utilizar funciones C para resolver problemas, recomendamos utilizar este enfoque al desarrollar programas. La modularidad resultante no sólo hace que el código sea más fácil de entender, sino que también mejora la capacidad de mantenimiento y fomenta la reutilización del código, lo cual es importante en el desarrollo de software. Al aplicar técnicas de programación modular, los desarrolladores pueden optimizar su productividad, reducir el riesgo de errores y crear sistemas más flexibles y adaptables. Este método es especialmente valioso en proyectos grandes donde la complejidad es un factor importante. En resumen, la integración eficaz de funciones al desarrollar programas en C es una práctica que contribuye significativamente a la calidad y el mantenimiento del código.

ENLACES DE LOS ARCHIVOS:

ENLACE GITHUB:

<https://github.com/JosueGuerra2023B/programacion2023B/tree/main/Laboratorio7%20%2002-01-2024>

ENLACE ONEDRIVE:

[Laboratorio7 02-01-2024](#)

PRESENTACIÓN

Al finalizar tu laboratorio deberás subir:

✦ Un archivo en formato pdf con el nombre (Laboratorio7_Programación_NApellido).

El laboratorio se puede hacer en parejas. Recuerde que más que presentar lo importante es que aprendan.

Bibliografía

- [1] deividcoptero, «Youtube,» 27 04 2013. [En línea]. Available:
] https://youtu.be/zMiNHqYQ9vw?si=9vWud6B_zh2jZZZT. [Último acceso: 31 12 2023].
- [2] L. Alberto, «Youtube,» 09 05 2016. [En línea]. Available: <https://youtu.be/A-NWAm0avps?si=3v8whiYPLuChSCo3>. [Último acceso: 31 12 2023].
- [3] I. Ivonne, «Aulas virtuales epn,» 22 12 2023. [En línea]. Available:
] https://aulasvirtuales.epn.edu.ec/pluginfile.php/9158054/mod_resource/content/0/Funciones.pdf. [Último acceso: 31 12 2023].