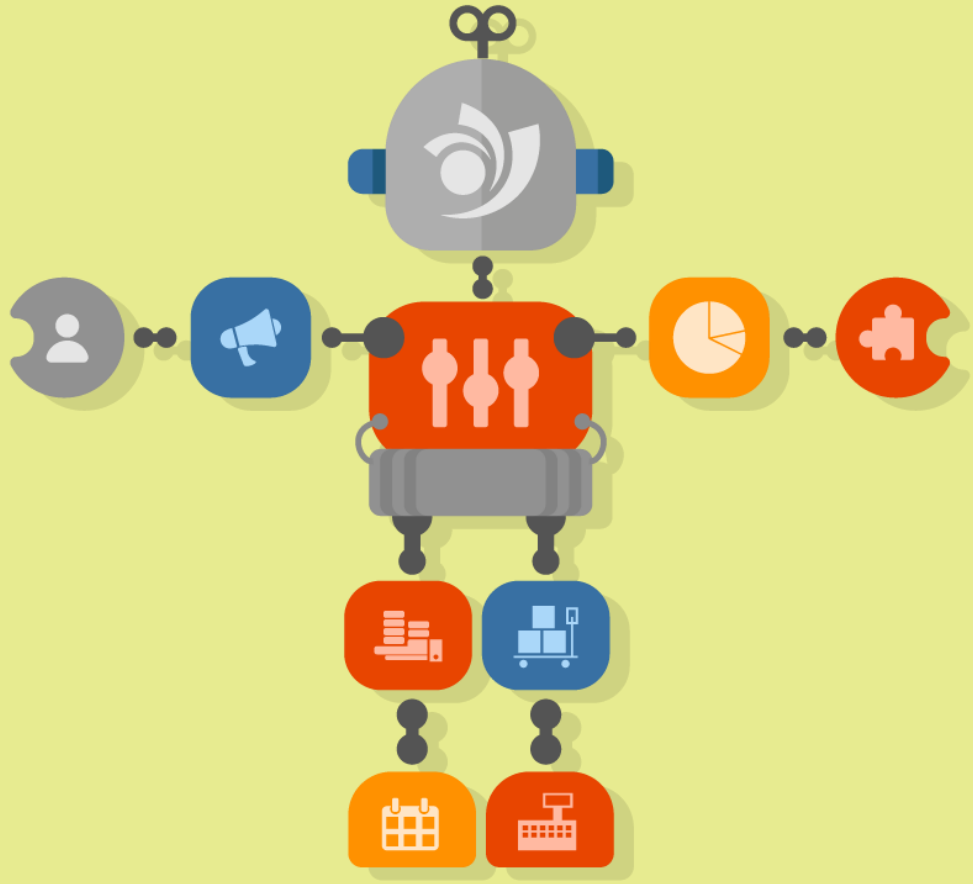
UNIVERSIDAD NACIONAL DE TRUJILLO
UNT

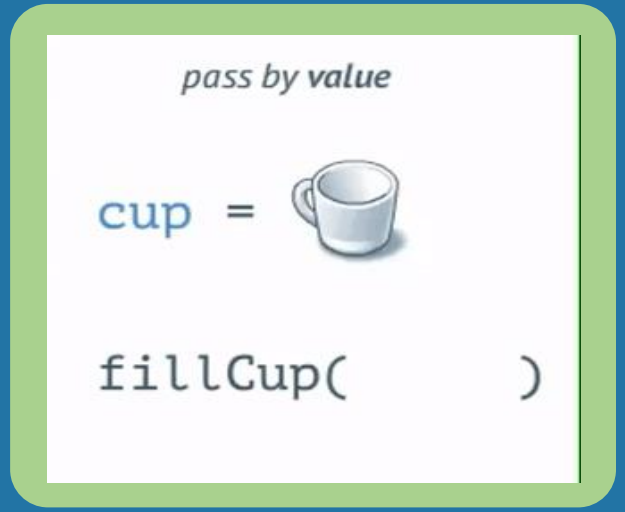
TIPOS DE PARÁMETROS DE LAS FUNCIONES



Mg. Marcelino Torres Villanueva

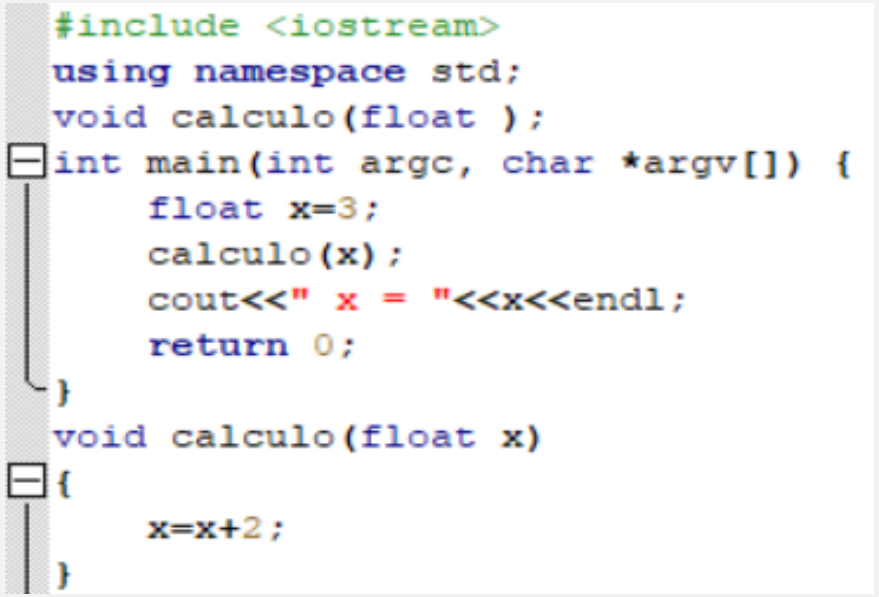
PARÁMETROS POR VALOR

Son aquellos a través de los cuales se pasan valores a la función, es decir se hace una copia de la variable pasada como argumento. A estos parámetros se les conoce como parámetros de entrada.



Ejemplo

	PSEUDOCODIGO
	calculo (float x) x ← x + 2 **fin_calculo** **Algoritmo** float x x ← 3 calculo (x) escribir x **fin_algotimo**

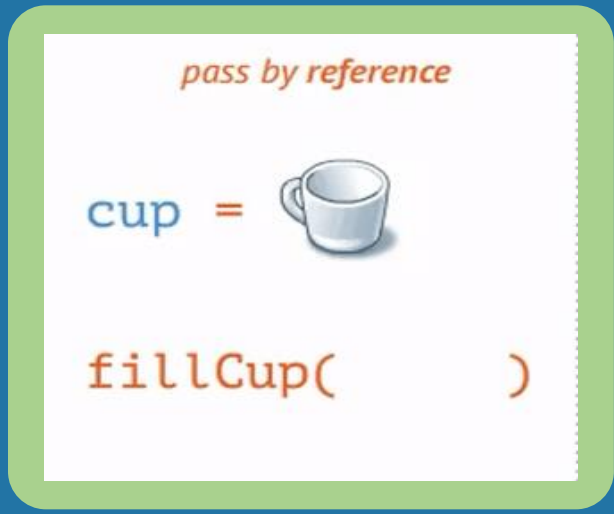
CODIFICACION	SALIDA DE PANTALLA
	El resultado es : x=3

```
#include <iostream>
using namespace std;
void calculo(float );
int main(int argc, char *argv[]) {
    float x=3;
    calculo(x);
    cout<<" x = "<<x<<endl;
    return 0;
}
void calculo(float x)
{
    x=x+2;
}
```

La variable x del programa principal al llamar a la función calculo hace una copia de su valor al parámetro de la función, luego la que se incrementa es la variable x de la función y no la variable x del programa principal.

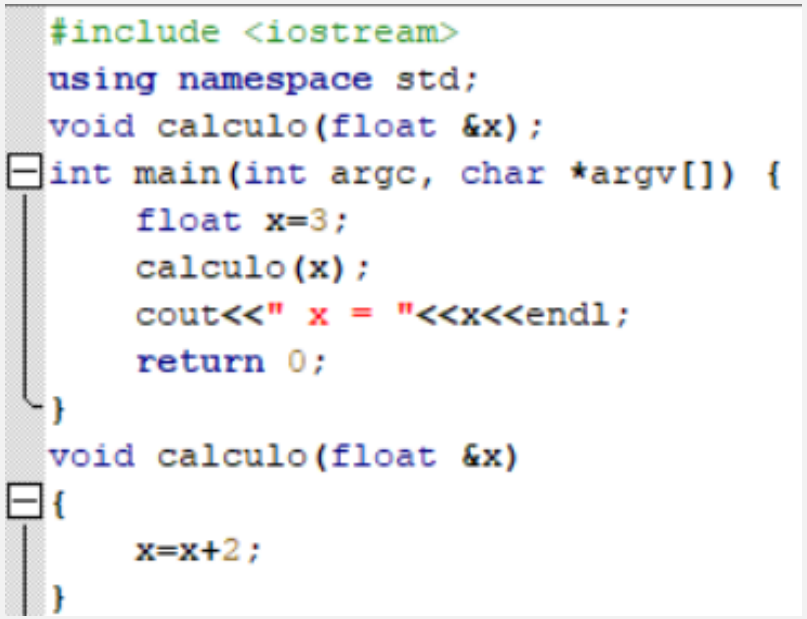
PARÁMETROS POR REFERENCIA

Son aquellos a través de los cuales se pasan referencias de las variables esto permite que las variables pasadas como argumento se puedan modificar. Para declarar un parámetro por referencia se utiliza el operador referencia &.



Ejemplo

	PSEUDOCODIGO
	calculo (real x(R)) x ← x + 2 **fin_calculo** **Algoritmo** float x x ← 3 calculo (x) escribir x **fin_algoritmo**

CODIFICACION	SALIDA DE PANTALLA
	El resultado es : x=5

```
#include <iostream>
using namespace std;
void calculo(float &x);
int main(int argc, char *argv[]) {
    float x=3;
    calculo(x);
    cout<<" x = "<<x<<endl;
    return 0;
}
void calculo(float &x)
{
    x=x+2;
}
```

La variable x del programa principal al llamar a la función calculo pasa la referencia de la variable esto hace que la variable x se pueda modificar.

EJERCICIOS RESUELTOS

Ejercicios resueltos

01. Programa para intercambiar el valor de 2 variables numéricas

PSEUDOCODIGO

```
cambio(real x(R), real y (R) )
    real t
    t ← x
    x ← y
    y ← t
fin_cambio
```

PSEUDOCODIGO

Algoritmo intercambio_variables

```
real a, b
escribir "Valor de a: "
leer a
escribir "Valor de b: "
leer b
cambio(a,b)
escribir "Nuevo valor de a: ", a
escribir "Nuevo valor de b: ", b
```

fin_algoritmo

CODIFICACION	SALIDA DE PANTALLA

```
#include<iostream>
using namespace std;
void cambio(float &, float &);
int main(int argc, char *argv[])
{
    float a,b;
    cout<<"Valor de a : ";
    cin>>a;
    cout<<"Valor de b : ";
    cin>>b;
    cambio(a,b);
    cout<<"Nuevo valor de a : "<<a<<endl;
    cout<<"Nuevo valor de b : "<<b<<endl;
    return 0;
}
void cambio(float &x, float &y)
{
    float t;
    t=x;
    x=y;
    y=t;
}
```

```
Valor de a : 75
Valor de b : 30
Nuevo valor de a : 30
Nuevo valor de b : 75
```

Ejercicios resueltos

02. Programa para ingresar el valor de un Punto en coordenadas Polares y reporte su equivalente en coordenadas cartesianas.

PSEUDOCODIGO

```
ingreso( real r (R), real t (R))
    hacer
        escribir "Valor de r: "
        leer r
    mientras r<=0
    escribir "Ángulo en grados sexagesimales: "
    leer t
fin_ingreso
```

PSEUDOCODIGO

```
calculo(real r, real t, real x(R), real y(R))
    t ← t * 3.1416/180
    x ← r * cos(t)
    y ← r * cos(t)
fin_calculo


reporte(real x, real y)
    escribir "x = ", x
    escribir "y = " , y
fin_reporte
```

PSEUDOCODIGO

Algoritmo coordenadas_Polares

```
real r, t, x, y
ingreso(r, t)
calculo(r, t, x, y)
reporte(x, y)
```

fin_algoritmo

CODIFICACION	SALIDA DE PANTALLA

```cpp
#include<iostream>
#include<math.h>
using namespace std;
void ingreso(float &,float &);
void calculo(float,float,float &,float &);
void reporte(float,float);
int  main(int argc, char *argv[])
{
    float r,t,x,y;
    ingreso(r,t);
    calculo(r,t,x,y);
    reporte(x,y);
    return 0;
}
void ingreso(float &r, float &t)
{
    do{
        cout<<"Valor de r : ";
        cin>>r;
    }while(r<=0);
    cout<<"Ángulo en grados sexagesimales : ";
    cin>>t;
}
void calculo(float r, float t, float &x, float &y)
{
    // convertimos el angulo de sexagesimales a radianes
    t=t*M_PI/180;
    x=r*cos(t);
    y=r*sin(t);
}
void reporte(float x, float y)
{
    cout<<"x = "<<x<<endl;
    cout<<"y = "<<y<<endl;
}
```

```
Valor de r : 90
Angulo en grados sexagesimales : 180
x = -90
y = -7.86805e-006
```

Ejercicios resueltos

03. Ingrese 2 puntos del plano cartesiano y reporte la ecuación de la recta que los contiene.

PSEUDOCODIGO

```
ingreso(real x1 (R), real y1 (R), real x2 (R), real y2(R))
    escribir "Primer punto"
    escribir "Valor de x: "
    leer x1
    escribir "Valor de y: "
    leer y1
    escribir "Segundo punto"
    escribir "Valor de x: "
    leer x2
    escribir "Valor de y: "
    leer y2
fin_ingreso
```

PSEUDOCODIGO

calculo(real x1, real y1, real x2, real y2, real m(R), real b(R))

m ← (y2 – y1) / (x2 – x1)
b ← y1 – m * x1

fin_calculo

PSEUDOCODIGO

```
reporte(real m, real b)

    escribir "y = ", m, "x"
    si b > 0 entonces
        escribir " + ", b
    sino
        escribir b
    fin_si

fin_reporte
```

PSEUDOCODIGO

Algoritmo puntos_recta

reales x1,y1,x2,y2, m, b

ingreso(x1, y1, x2, y2)

calculo(x1, y1, x2, y2, m, b)

reporte(m, b)

fin_algoritmo

CODIFICACION

```
#include<iostream>
#include<math.h>
using namespace std;
void ingreso(float &,float &,float &, float &);
void calculo(float,float,float,float,float &,float &);
void reporte(float,float);
int main(int argc, char *argv[])
{
    float x1,y1,x2,y2,m,b;
    ingreso(x1,y1,x2,y2);
    calculo(x1,y1,x2,y2,m,b);
    reporte(m,b);
    return 0;
}
void ingreso(float &x1, float &y1, float &x2, float &y2)
{
    cout<<"Primer Punto"<<endl;
    cout<<"Valor de x : ";
    cin>>x1;
    cout<<"Valor de y : ";
    cin>>y1;
    cout<<"Segundo Punto"<<endl;
    cout<<"Valor de x : ";
    cin>>x2;
    cout<<"Valor de y : ";
    cin>>y2;
}
void calculo(float x1, float y1, float x2, float y2,float &m,float &b)
{
    m=(y2-y1)/(x2-x1);
    b=y1-m*x1;
}
void reporte(float m, float b)
{
    cout<<"y = "<<m<<"x";
    if(b>0)
        cout<<"+"<<b<<endl;
    else
        cout<<b<<endl;
}
```

SALIDA DE PANTALLA

```
Primer Punto
Valor de x : 10
Valor de y : 12
Segundo Punto
Valor de x : 15
Valor de y : 17
y = 1x+2
```

03. Reportar los n primeros números primos

PSEUDOCODIGO

```
leeNro(entero n(R))

    Hacer
        Escribir "Valor de n: "
        Leer n
    mientras n<=0

fin_leeNro
```

PSEUDOCODIGO

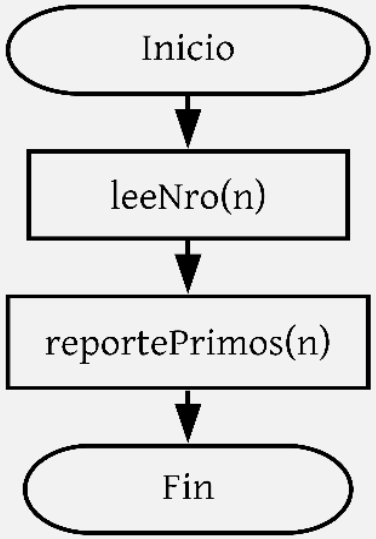
```
logico esPrimo(entero n)

    entero i, cd
    cd ← 0
    para i ← 1 hasta n inc 1 hacer
        si  n mod i = 0 entonces
            cd ← cd +1
        fin_si
    fin_para
    si cd = 2 entonces
        retornar verdadero
    sino
        retornar falso
    fin_si

fin_esPrimo
```

PSEUDOCODIGO

reportePrimos(entero n)

```
entero c
entero num
c ← 0
num ← 0
mientras c < n hacer
    num ← num + 1
    si esPrimo(num) entonces
        escribir num
        c ← c +1
    fin_si
fin_mientras
```

fin_reportePrimos

DIAGRAMA DE FLUJO	PSEUDOCODIGO
	**Algoritmo reportePrimos** leeNro(n) reportePrimos(n) **fin_algoritmo**

SALIDA DE PANTALLA

Valor de n: 10
2 3 5 7 11 13 17 19 23 29

CODIFICACION

```cpp
#include<iostream>
using namespace std;
void leeNro(int &n);
bool esPrimo(int n);
void reportePrimos(int n);
int main(int argc, char *argv[])
{
    int n;
    leeNro(n);
    reportePrimos(n);
    return 0;
}
void leeNro(int &n)
{
    do{
        cout<<"Valor de n: ";
        cin>>n;
    }while(n<=0);
}
bool esPrimo(int n)
{
    int i,cd=0;
    for(i=1;i<=n;i++)
    {
        if(n % i==0)
            cd++;
    }
    if(cd==2) return true;
    else return false;
}
```

```cpp
bool esPrimo(int n)
{
    int i,cd=0;
    for(i=1;i<=n;i++)
    {
        if(n % i==0)
            cd++;
    }
    if(cd==2) return true;
    else return false;
}
void reportePrimos(int n)
{
    int c=0,num=0;
    while(c<n)
    {
        num++;
        if(esPrimo(num))
        {
            cout<<num<<" ";
            c++;
        }
    }
    cout<<endl;
}
```

Sobrecarga de Funciones

Sobrecarga de una función es usar el mismo nombre para diferentes funciones, distintas unas de otras por sus listas de parámetros.

En realidad, la sobrecarga de funciones es una propiedad que facilita la tarea al programador cuando se desean diseñar funciones que realizan la misma tarea general pero que se aplican a tipos de parámetros diferentes. Estas funciones se pueden llamar sin preocuparse sobre cuál función se invoca ya que el compilador detecta el tipo de dato de los parámetros y ejecuta la función asociada a ellos.

```cpp
#include <iostream>
using namespace std;

void imprimir(int e);
void imprimir(double f);
void imprimir(char c);
void imprimir(bool b);

int main(int argc, char *argv[]) {
    imprimir(12);
    imprimir(6.5);
    imprimir('q');
    imprimir(true);
    return 0;
}

void imprimir(int e)
{
   cout<<"El valor es : "<<e<<endl;
}
void imprimir(double f)
{
   cout<<"El valor es : "<<f<<endl;
}

void imprimir(char c)
{
   cout<<"El valor es : "<<c<<endl;
}
void imprimir(bool b)
{
    cout<<"El valor es : "<<b<<endl;
}
```

```cpp
#include <iostream>
using namespace std;

float promedio(float x1, float x2);
float promedio(float x1, float x2, float x3);


int main(int argc, char *argv[]) {

    cout<<"Promedio : "<<promedio(3,5)<<endl;
    cout<<"Promedio : "<<promedio(8,10,15)<<endl;
    return 0;
}

float promedio(float x1, float x2)
{
    return (x1+x2)/2;
}
float promedio(float x1, float x2, float x3)
{
    return (x1+x2+x3)/3;
}
```

```cpp
#include <iostream>
using namespace std;

int cuadrado (int n);
float cuadrado(float n);
double cuadrado(double n);

int main(int argc, char *argv[]) {
    cout<<cuadrado(2)<<endl;
    cout<<cuadrado(3.5f)<<endl;
    cout<<cuadrado(2.8)<<endl;
    return 0;
}

int cuadrado (int n)
{
    return n*n;
}
float cuadrado(float n)
{
    return n*n;
}
double cuadrado(double n)
{
    return n*n;
}
```

Parámetros con valores por defecto

C++ permite tener valores por defecto para los parámetros. Esto supone que, si no se pasa el parámetro correspondiente, se asume un valor predefinido.

La forma de indicarlo es declararlo en el prototipo de la función.

Ejemplo:

```
float suma(float a, float b=3, float c=5);  // o tambien

float suma(float, float =3,float =5);
```

La gramática de C++ exige que los parámetros con valores por defecto deben ser los últimos en la lista de parámetros, y que si en una ocasión falta algún argumento, los que le siguen también deben faltar (adoptar también los valores por defecto).

```cpp
#include <iostream>
using namespace std;

float suma(float a, float b=4, float c=5);

int main(int argc, char *argv[]) {
    cout<<"La suma es : "<<suma(12,10,15)<<endl;
    cout<<"La suma es : "<<suma(3,7)<<endl;
    cout<<"La suma es : "<<suma(2)<<endl;
    return 0;
}

float suma(float a, float b, float c)
{
   return a+b+c;
}
```