

Repaso de Informática 1 - Parte 1

Ing José Luis MARTÍNEZ

27 de marzo de 2020

1. Ejercicios

. La presente lista de ejercicios resueltos sirven de referencia para practicar. Tenga presente que en algunos casos deberá estar atento para efectuarles modificaciones para lograr una adecuada compilación.

1. Escribir un programa en C, que permita imprimir en la pantalla el típico mensaje introductorio en todos los lenguajes de programación: "Hola Mundo".

```
#include <stdio.h>
/*es igual que "stdio.h", difieren en donde se comienza a buscar el archivo .h*/
int main(void) {
    printf("Hola mundo\n"); }
```

2. Idem anterior, utilizando una función para ejecutar la misma acción.

```
#include <stdio.h>
int main( ) {
    hola ( );
    getchar( );
    return 0; }
hola ( )
{ printf("Hola mundo\n");
}
```

3. DUP que permita imprimir en la pantalla un mensaje con el nombre de esta Universidad.

```
#include <stdio.h>
int main( )
{ printf("Universidad Tecnológica Nacional\n");
return 0; }
```

0-4) DUP que permita calcular el cuadrado de un número dado como dato, utilizando una función.

```
#include <stdio.h>
#include <matemat.h>
int main( ){
    int num;
    num=100; /*100 es el número dado*/
```

```
sqr(num); /*se invoca sqr( ) con num como argumento*/
getchar( );
return 0; }
sqr(int x) /*en este caso x -parámetro formal- recibe lo que envía num*/
{ printf("%d al cuadrado es %d\n",x,x*x);}
```

4. DUP que permita imprimir en la pantalla, la edad de quien escribe.

```
#include <stdio.h>
int main( ){
int edad;
printf("Ingrese su edad:");
scanf("%d",&edad);
printf("Mi edad es %d años\n",edad);
getchar( );
return 0; }
```

5. DUP que permita jugar intentando descubrir un número desconocido, admitiendo el ingreso de propuestas por parte del jugador desde el teclado.

```
#include <stdio.h>
#include<stdlib.h> /*por rand( ) */
int main( ){
int aleatorio;
int intento;
aleatorio=rand( )%10 /*de 0 a 9*/;
printf("adivine el numero en juego:");
scanf("%d",&intento);
if (intento==aleatorio) /*error si pongo = en vez de ==*/
{printf("correcto,%d es el nro. en juego\n",aleatorio);
}
getchar( );
return 0; }
```

6. Perfeccionar el programa anterior incorporando opciones que permitan enviar mensajes en caso de error o aproximación.

```
#include <stdio.h>
#include<stdlib.h>
#include<time.h> /*#include <sys/time.h */
int main( ){
int aleatorio, intento;
srand(time(0)); //genera valor de referencia basado en el reloj del sistema
aleatorio=rand( ) %10; /*de 0 a 9*/;
printf("adivine el numero en juego:");
scanf("%d",&intento);
if (intento==aleatorio) /*Cuidado: error si pongo = en vez de ==*/
{
printf("correcto,%d es el nro. en juego\n",aleatorio);
}
else {
printf("Incorrecto ");
}
```

```
if (intento>aleatorio)
printf("demasiado alto,el número era %d\n",aleatorio);
else printf("demasiado bajo,el número era %d\n",aleatorio);
}
getchar ( );
return 0; }
```

7. DUP que permita efectuar la transformación de unidades de medida dadas en valores enteros de pies a metros lineales.

```
#include <stdio.h>
int main( ){
int pies;
float metros; /*separadas por comas las variables de punto flotante*/
printf("Introduzca el número de pies: ");
scanf("%d",&pies); /*se lee un entero*/
metros=pies *0.3048; /*fórmula de conversion de pies a metros*/
printf("%d pies son %f metros\n",pies,metros);
getchar( );
return 0; }
```

8. Reescribir el programa anterior para que permita efectuar la transformación de unidades de medida dadas en punto flotante para los pies.

```
#include <stdio.h>
int main( ){
float pies,metros; /*separadas por comas las variables*/
printf("Introduzca el número de pies: ");
scanf("%f",&pies); /*se lee un float*/
metros = pies *0.3048; /*fórmula de conversion de pies a metros*/
printf("%f pies son %f metros\n",pies,metros);
getchar( );
return 0; }
```

9. DUP que permita multiplicar dos números dados.

```
# include <stdio.h>
int main ( )
{
int respuesta;
respuesta = mul(3,4); /*en mul se ponen los argumentos*/
printf("La respuesta es %d\n", respuesta);
getchar( );
return 0; }

mul(int a, int b)
{ return a*b; /*retorna un valor */}
```

10. DUP que calcule el cociente y el resto de un número entero que se introduce desde el teclado.

```
#include <stdio.h>
int main( ){
    int x,y;
    printf("Introduzca el dividendo y el divisor:");
    scanf("%d%d",&x,&y);
    printf("El cociente es",x/y); /*El operador % permite hacer la división módulo que es dist.
    getchar( );
    return 0; }
```

11. DUP que permita imprimir los números pares entre 1 y 100.

```
#include <stdio.h>
int main( ){
    int i;
    for (i=1;i<=100;i++) /*! es el operador lógico NOT, que invierte el resultado*/
    if(!(i%2)) /*la operación % módulo da falso=0, cuando se utilice un número par*/
    printf("%d",i);
    getchar( );
    return 0; }
```

12. DUP que permita calcular el producto de dos números, y verificar si uno de ellos es negativo.

```
#include <stdio.h>
int main( ){
    int x,y, producto;
    printf("Introduzca dos números:");
    scanf("%d%d",&x,&y);
    if((producto= x * y) < 0)
    printf("Uno de los números es negativo\n");
    else
    printf("El producto positivo es %d", producto);
    getchar( );
    return 0; }
```

13. DUP que permita visualizar el resultado de la aplicación de distintos códigos de barra invertida dentro de printf para la operación de división entre dos números enteros (Ej: 99/2 y enero)

```
#include<stdio.h>
void main( ){
    int i, j;
    printf("Introduzca dos números:");
    scanf("%d%d", &i,&j);
    printf("%d ", i/ j); /*d: número base diez=49*/
    printf("%e", i/ j); /*e: exponencial4.95+01*/
    printf("%f", i/j); /*f: punto flotante =49.5000*/
    printf("%.2f", i/ j); /*f: punto flotante =49.50*/
    printf("%g", i/j); /*g: en d y e elimina ceros=49.5*/
    printf("%o", i); /*o: octal =143*/
    printf("%x", i); /*x: número hexadecimal s/signo=63*/
    printf("%s", "enero"); /*s: cadena=enero*/}
```

14. DUP que calcule e imprima, las raíces cuadradas de los 15 primeros números enteros.

```
#include <stdio.h>
#include <math.h> /*inclusión de librería con funciones matemáticas*/
int main( ) {
    int i;
    for (i=1; i<16; i=i+1)
    {
        printf("La raíz cuadrada de %2d es = a %6.4f \n", i, sqrt(i));
    }
    getchar( );
    return 0; }
```

15. Reescribir el programa anterior utilizando la sentencia while.

```
#include<stdio.h>
#include<math.h>
int main( ) {
    int i; /*declaración de variable:nombre y tipo*/
    i = 1; /*asignación de valor inicial a la variable*/
    while (i<16)
    {
        printf("La raíz cuadrada de %2d es = a %6.4f\n",i,sqrt(i));
        i++;
    }
    getchar( );
    return 0; }
```

16. DUP que permita efectuar la conversión de temperaturas de grados fahrenheit a grados celsius.

```
#include<stdio.h>
int main( ) {
    int fahr,celsius,lower,upper,step;
    lower=0;
    upper=140;
    step=20;
    fahr=lower;
    while (fahr<=upper){
        celsius=5*(fahr-32)/9; /*fórmula de equivalencia*/
        printf("%d\t%d\n",fahr,celsius);
        fahr=fahr+step; }
    getchar( );
    return 0; }
```

17. Modificar el programa anterior para que permita la manipulación de datos decimales.

```
#include<stdio.h>
int main( ) {
    float fahr,celsius;
    int lower,upper,step;
    lower=0;
```

```
upper=140;
step=20;
fahr=lower;
while (fahr<=upper) {
celsius=(5.0*(fahr-32.0))/9.0;
printf("%3.0f%6.1f\n",fahr,celsius);
fahr=fahr+step; } getchar( );
return 0; }
```

18. Diseñar el mismo programa pero utilizando un bucle FOR.

```
#include<stdio.h>
int main( ) {
int fahr;
for(fahr=0;fahr<=300;fahr=fahr+20)
printf("%3d %6.1f\n",fahr,(5.0/9.0)*(fahr-32.0));
getchar( );
return 0; }
```

19. DUP que muestre en pantalla los múltiplos de 2 entre 0 y 40.

```
#include<stdio.h>
int main( ) {
int mult, tope, cont;
mult=0;
cont=0;
tope=40;
while (mult < tope) {
mult = cont *2;
printf ("%4d\n", mult);
++cont;
}
getchar( );
return 0; }
```

20. DUP que capture una cadena de caracteres de la entrada y la imprima en la pantalla.

```
#include<stdio.h>
int main( ) {
char cadena;
while ((cadena =getchar ( )) != '\n'){
putchar (cadena);
}
getchar( );
return 0; }
```

21. DUP que imprima en pantalla el número de caracteres tecleados y el número de líneas empleadas hasta que se pulse Ctrl Z".

```
#include<stdio.h>
int main( ) {
int cont, lin, ca;
cont =0;
lin=0;
ca= 0;
while ((ca =getchar ( )) != EOF){
++ cont;
if (ca == '\n' {
++ lin; }
}
printf ("%2d \n", EOF);
printf ("%3d caracteres tecleados. \n", cont);
printf ("En %2d líneas. \n", lin);
getchar( );
return 0; }
```

22. Obtener una tabla del código ASCII.

```
#include<stdio.h>
int main( ) {
int x;
char = ca;

for (x = 32; x <= 255; ++x) {
ca = x;
printf ("%d...%c/ \t", x, ca);
}
getchar( );
return 0; }
```

23. Completar esta función para que permita la conexión de un usuario a un determinado sistema. Luego DUP que la utilice.

```
void conexión (void) {
char password [20];
int x;
for (x = 0; x <3 && strcmp(password, "mi_contraseña"); ++x) /*devuelve 0 si son iguales*/
{ printf ("Por favor, introduzca su clave: ") ;
gets (password); }
if (x ==3 ) return;
else (strcmp (password, "mi_contraseña")) { /*da falso=0 si son iguales*/
printf("Contraseña incorrecta\n");
return 0; }
return 1; /* si las palabras son iguales*/
```

2. Problemas Propuestos

1. En la página web <https://docs.microsoft.com/en-us/cpp/c-language/cpp-integer-limits?view=vs-2019>, se encuentran los valores límites para cada tipo de variable. DUP que muestre los valores límites de su computadora. Por ejemplo:

```
printf("El minimo valor de un int es: ", INT_MIN);
```

2. DUP que visualice la aplicación de los distintos operadores relacionales sobre dos enteros ingresados por teclado. (operador == igual; operador != distinto; operador <= menor o igual que; operador < mayor o igual que; operador <= menor que; operador > mayor que).

```
/*las expresiones relacionales y lógicas producen un resultado que es solamente 1 -cierto-
```

3. DUP que imprima en pantalla el número de caracteres y el número de líneas ingresados por teclado. La ejecución debe concluir al pulsar las teclas Control+Z".
4. DUP que determine, de un vector de cinco elementos con los números 20, 21, 22, 23, 24, en que elemento del vector se guarda cada número y en que posición de memoria se halla cada uno, mostrando en la pantalla el resultado.
5. DUP que determine si un número introducido por teclado es igual a uno cualquiera de los elementos de un vector determinado, cuyo contenido es 20, 6, 7, 25, 49, 28.
6. DUP un programa que tomados los valores de una ecuación de segundo grado saque sus raíces, teniendo en cuenta que pueden ser imaginarias
7. DUP un programa en C que , al recibir como dato una X cualquiera, calcule el cos(x) utilizando la siguiente serie:

$$\cos x = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \dots$$

La diferencia entre la serie y un nuevo término debe ser menor o igual a 0,001. Imprima el número de términos requerido para obtener esta precisión

Ayuda: La serie queda totalmente representada como

$$\sum_{k=0}^{\infty} \frac{(-1)^k \cdot x^{2k}}{2k!}$$

8. Un número N es primo si los únicos enteros positivos que lo dividen son exactamente 1 y N. Escribe un programa en C que , al recibir como dato un número entero positivo, determine si este es un número primo.
9. Se dice que un número es considerado perfecto si la suma de sus divisores excepto el mismo, es igual al propio número. Escriba un programa que obtenga e imprima todos los números perfectos comprendidos entre 1 y N.
10. Escriba una función que dado cualquier número N, imprima sus divisores primos.
11. Escriba un programa en C que, al recibir como dato un arreglo de números enteros:
- a) Lo imprima en forma inversa
 - b) Lo ordene de mayor a menor
 - c) Lo ordene de menor a mayor
12. DUP que, al recibir un arreglo unidimensional de tipo real que contiene mediciones, calcule lo siguiente:
- a) La media aritmética.
 - b) La varianza.
 - c) La desviación estándar.

- d) La moda. Se calcula obteniendo el número con mayor frecuencia.
 - e) La mediana
 - f) Optimizar los resultados mediante una regresión lineal de mínimos cuadrados. (Si no los vio en Física I, solicite al docente las ecuaciones)
13. Diseñe un programa que dado un arreglo unidimensional, determine si existe un elemento del arreglo.
- a) Realizando una búsqueda secuencial
 - b) Realizando una búsqueda binaria

3. Anexo

3.1. Ingreso de caracteres especiales en Linux.

La forma de insertar caracteres especiales, no es en formato ASCII sino Unicode.

-Una manera de insertarlos es con la combinación:

CTRL+SHIFT+u+(código Unicode)

El código Unicode es de cuatro números en hexadecimal manteniendo presionadas continuamente sólo las teclas CTRL y SHIFT(mayúsculas), y tipeando las demás. Al soltar CTRL y SHIFT, el carácter se insertará.

Para conocer el código Unicode de un carácter, se puede ejecutar el Mapa de caracteres (desde el menú Aplicaciones de Gnome o con el comando charmap, ó Menú – > Accesorios – > Mapa de Caracteres), buscar el carácter deseado y fijarse en la barra de estado que va a aparecer algo como Ü+004A LATIN CAPITAL LETTER J”(por ejemplo para la letra J). Esos números después del U+ son los que deben escribir con la combinación de teclas anteriores.

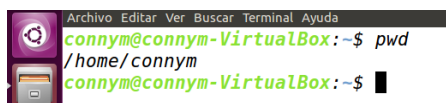
-Otra forma es copiando los caracteres deseados desde el mismo Mapa de caracteres.

-También Utilizando [Alt Gr], como en Windows: Utilice este método para obtener:

[Alt + Gr]+[Ñ] = ~
[Alt + Gr]+[Z] = <<
[Alt + Gr]+[X] = >>
[Alt + Gr]+[V] = ”
[Alt + Gr]+[B] = ”
[Alt + Gr]+[,] = guión largo

3.2. Comandos basicos para crear un programa en Linux

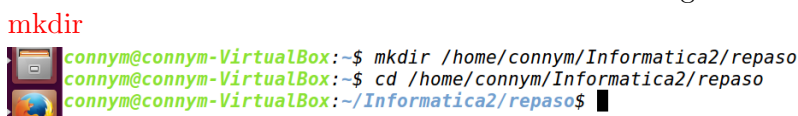
1. Mediante el comando **pwd** averiguamos en cual directorio estamos ubicados



```
Archivo Editar Ver Buscar Terminal Ayuda
connym@connym-VirtualBox:~$ pwd
/home/connym
connym@connym-VirtualBox:~$
```

2. A continuación creamos el directorio donde vamos a guardar nuestro programa con el comando

mkdir



```
connym@connym-VirtualBox:~$ mkdir /home/connym/Informatica2/repaso
connym@connym-VirtualBox:~$ cd /home/connym/Informatica2/repaso
connym@connym-VirtualBox:~/Informatica2/repaso$
```

3. Utilizamos el editor de texto `gedit`

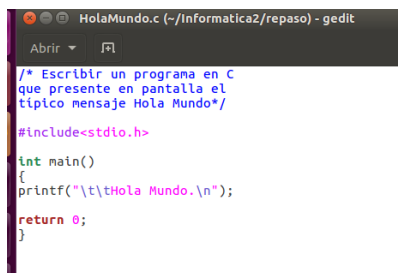


```
connym@connym-VirtualBox:~$ cd /home/connym/Informatica2/repaso
connym@connym-VirtualBox:~/Informatica2/repaso$ gedit HolaMundo.c
```

4. Se abre la ventana del programa



5. Escribimos nuestro programa



6. Para compilar el programa volvemos al terminal y ejecutamos la siguiente orden

```
connym@connym-VirtualBox:~/Informatica2/repaso$ gcc HolaMundo.c -o HolaMundo
connym@connym-VirtualBox:~/Informatica2/repaso$ ls
HolaMundo  HolaMundo.c
```

7. Vemos que al ejecutar el comando `ls` para que muestre el contenido del directorio, ahora también está presente el archivo `HolaMundo`, que va a ser nuestro archivo ejecutable.

Para correr el programa se antepone `./` al nombre del archivo.

```
connym@connym-VirtualBox:~/Informatica2/repaso$ ./HolaMundo
    Hola Mundo.
```

8. El comando `cat` permite ver el interior del archivo

```
connym@connym-VirtualBox:~/Informatica2/repaso$ cat HolaMundo.
/* Escribir un programa en C
que presente en pantalla el
típico mensaje Hola Mundo*/

#include<stdio.h>

int main()
{
    printf("\t\tHola Mundo.\n");
    return 0;
}
```

9. Cuando utilizamos funciones de la biblioteca matemática, puede suceder que al compilar no encuentra la función `sqrt()` por ejemplo. Para solucionar este error se debe agregar `-lm` al final de la instrucción de compilación.

```
~/Informatica2/repaso$ gcc ej_0_16.c -o ej_0_16 -lm
```