MATERIA: Programación I
GUIA DE LABORATORIO #: 18, 19
Fecha:

COMPETENCIAS:

En esta guía se realizará la descripción y aplicaciones de los siguientes elementos de programación:

1. Contadores
2. Acumuladores
3. Conversión de tipos de datos (cast)

TEORÍA ASOCIADA:

1. CONTADORES

En programación, un contador es una variable que modifica su valor en una cantidad constante k, que generalmente es 1, cada vez que se presenta un evento a una situación particular a tomar en cuenta.

La modificación puede incrementar o decrementar el valor de la variable contadora.

Es usual que la variable contadora se inicialice en 0.

La expresión general de una variable contadora que se modifica en una cantidad k, es:

```
variable = variable + k;
```

Cuando k es 1 (o -1), es conveniente usar los siguientes operadores abreviados:

```
variable++;   // post incremento, se usa la variable y posteriormente se incrementa.
variable--; // post decremento, se usa la variable y posteriormente se decrementa.
++variable; // pre incremento, se incrementa la variable y después se la usa.
--variable; // pre decremento, se decrementa la variable y después se la usa.
```

Para ilustrar las diferencias de los operadores abreviados, considere las diferencias de las salidas de los dos códigos siguientes:

```
int a=2, b=5;
cout<<a++<<" "<<++b<<endl;     // 2 6
cout<<a<<" "<<b<<endl;         // 3 6

int a=2, b=5;
cout<<++a<<" "<<b++<<endl;     // 3 5
cout<<a<<" "<<b<<endl;         // 3 6
```

2. ACUMULADORES

Un acumulador, o totalizador, es una variable que modifica su valor en una cantidad no constante, generalmente positiva, cada vez que se presenta un evento o una situación particular.

La expresión general de la variable acumuladora es:

```
variable = variable + k;
```

en la que k puede adoptar diferentes valores a lo largo de la ejecución de un programa.

Las variables contadoras y acumuladoras se usan en muchísimas ocasiones en los sistemas informáticos, como en el control de las ventas del día, inventarios y stocks en almacenes, los gastos realizados en alguna institución, cuentas bancarias, etc., etc.

3. CONVERSIÓN DE TIPOS DE DATOS (cast)

La conversión de tipos de datos, permite cambiar el tipo de dato del valor de una literal, contenido de una variable o el resultado expresión, a otro tipo de dato del que tiene originalmente.

Se distinguen dos casos particulares de conversión:

- ✓ conversión implícita
- ✓ conversión explícita

La conversión o cast implícita la realiza el compilador automáticamente, sin necesidad de ninguna especificación por parte del programador; esta conversión se realiza cuando no existe la posibilidad de modificación del valor original en el tipo de dato destino o cuando no hay pérdida de precisión en la conversión; por ejemplo, en las instrucciones siguientes, el tipo de dato de los valores de la derecha automáticamente y de forma correcta se convierten en los tipos de datos especificados en la declaración de las variables a la izquierda:

```
float a = 123;
double b = 2345.5f;
long long c = 98434;
bool d = 25;
char e = 65;
```

El cast explicito se usa cuando existe la posibilidad de que el valor del tipo de dato de origen pueda ser alterado o provocar una pérdida de precisión en el valor destino; el compilador no restringe la conversión, pero al existir ese riego, es el programador que debe asumir esa responsabilidad, lo que se realiza especificando explícitamente por medio de uno de las dos sintaxis siguientes:

- ✓ (tipoDestino) valor; sintaxis cast del lenguaje C
- ✓ tipoDestino (valor); sintaxis del lenguame C++

por ejemplo:

```
int a = (int) 123.34;
float b = float (2.34e1);
short c = (short) 23452L;
char d = char (122);
```

La alteración del valor en el destino (izquierda) se presenta cuando el rango del tipo de dato destino no contiene al valor de origen (derecha), como en los siguientes casos:

```
int a = (int) 12.234e50;
float b = float (2.34e300);
short c = (short) 2345223472834L;
char d = char (1224344);
```

EJERCICIOS RESUELTOS:

Problema1:
1. Escribir un programa que lea un número natural: n (0 < n <= 80), e imprima un triángulo rectángulo de base y alto igual a n, alineado a la derecha, formado por los dígitos del 0 al 9 de forma consecutiva y cíclica, como se muestra en el ejemplo que sigue:

EJEMPLO DE ENTRADA
6
EJEMPLO DE SALIDA

```
     0
    12
   345
  6789
 01234
567890
```

```
#include <iostream>
using namespace std;
main()
{
   int n, dato=0;
   cin>>n;
   for(int i=0; i<n; i++)
   {
      for(int j=n-1; j>=0; j--)
      {
         if(j<=i)
         {
            cout<<dato;
            dato=++dato%10;
         }
         else
            cout<<" ";
      }
```

```
        cout<<endl;
    }
}
```

Problema2:
En el área de estadística y probabilidades es interesante comprobar que los valores teóricos de ciertos ensayos se cumplan en la práctica; desafortunadamente el elevado número de ensayos necesarios para tener resultados confiables impide la experiencia práctica; sin embargo, puede emplearse programas de computadora para simular los ensayos prácticos, de esta manera se han resuelto miles de problemas cuya solución teórica podría ser sumamente complicada, como en el proyecto Manhattam. En esta ocasión se desea verificar que la probabilidad de que salga "escudo" al lanzar una moneda sea 0.5. Escribir un programa que simule el lanzamiento de una moneda n veces e imprima la probabilidad calculada de que salga "escudo". El programa deberá leer un número natural: n (1e3 <= n <= 1e8) simular el lanzamiento de una moneda n veces e imprimir la probabilidad calculada con todos los decimales. Recordar que la probabilidad es igual a: (número de ensayos exitosos) / (número total de ensayos realizados).

EJEMPLO DE ENTRADA
100000000
EJEMPLO DE SALIDA
0.5

```
#include <iostream>
using namespace std;
main()
{
    int n, exitos=0;
    double prob=0;
    cin>>n;
    for(int i=0; i<n; i++)
        if(rand()%2==0)
            exitos++;
    prob=(double)exitos/n;
    cout<<prob<<endl;
}
```

Problema3:
Es bien sabido que un número natural: n, es múltiplo del 9 cuando se aplica la expresión: n%9==0 y el resultado es: true; pero antes de la llegada de las calculadoras, se usaba la técnica de la suma de los dígitos del número n, tal que al obtener 9 o un múltiplo de 9, se concluía que n era múltiplo de 9.
Escribir un programa que lea un número natural: n (0 < n <= 9e18) y realice la suma de sus dígitos y repetir ese procedimiento con el resultado, hasta obtener un resultado final de un solo dígito, luego determinar e imprimir si n es múltiplo de 9 o no lo es.

EJEMPLO DE ENTRADA

29097
sumatoria de los dígitos de un número
29097, es múltiplo de 9.

EJEMPLO DE ENTRADA
7922059822227890
sumatoria de los dígitos de un número
7922059822227890, no es múltiplo de 9.

```
#include <iostream>
#include <locale>
using namespace std;
int main()
{
    setlocale(LC_ALL, "");
    long long n, nn;
    int suma=0;
    cin>>n;
    nn=n;
    while(n>9)
    {
        while(n>0)
        {
            suma=suma+n%10;
            n=n/10;
        }
        n=suma;
        suma=0;
    }
    if(n==9)
        cout<<nn<<", es múltiplo de 9."<<endl;
    else
        cout<<nn<<", no es múltiplo de 9."<<endl;
    return 0;
}
```

Problema4:
La Real Academia Española, denomina número capicúa a aquel número que se lee igual de izquierda a derecha como de derecha a izquierda, por ejemplo, el 2073702 o el 4554. Escribir un programa que lea un número natural: n (0 <= n < 9e18), e indique si éste es capicúa o no lo es.

EJEMPLO DE ENTRADA
203498894302
EJEMPLO DE SALIDA
203498894302, es capicúa.

EJEMPLO DE ENTRADA

1234567890
EJEMPLO DE SALIDA
1234567890, no es capicúa.

```
#include <iostream>
#include <locale>
using namespace std;
main()
{
    setlocale(LC_ALL, "");
    int n, copia, inv=0;
    cin>>n;
    copia=n;
    while(copia>0){
        inv=10*inv+copia%10;
        copia=copia/10;
    }
    if(n==inv)
        cout<<n<<", es capicúa."<<endl;
    else
        cout<<n<<", no es capicúa."<<endl;
}
```

Problema5:
Escribir un programa que convierta un color, expresado en el formato hexadecimal a su equivalente RGB; para ello, el programa debe leer una cadena con el formato: #rrggbb, donde rr, gg y bb son pares de dígitos hexadecimales que deben ser convertidos, cada par, a su número en base 10 equivalente y deberán ser imprimidos en ese orden.

EJEMPLO DE ENTRADA
#0C2DC8
EJEMPLO DE SALIDA
12 45 200

EJEMPLO DE ENTRADA
#ff15bb
EJEMPLO DE SALIDA
255 21 187

```
#include <iostream>
#include <string>
using namespace std;
main()
{
    string colorHex, hex="0123456789ABCDEF";
    cin>>colorHex;
    int rr=16*hex.find(toupper(colorHex.at(1)));
    rr+=hex.find(toupper(colorHex.at(2)));
```

```
    int gg=16*hex.find(toupper(colorHex.at(3)));
    gg+=hex.find(toupper(colorHex.at(4)));
    int bb=16*hex.find(toupper(colorHex.at(5)));
    bb+=hex.find(toupper(colorHex.at(6)));
    cout<<rr<<" "<<gg<<" "<<bb<<endl;
}
```

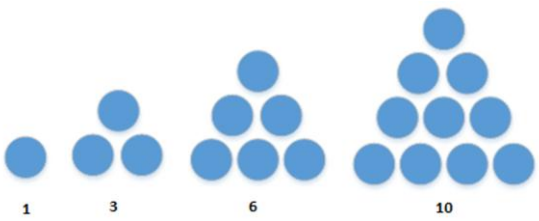
Problema6:
Las calculadoras y computadoras usan series para calcular los resultados de diferentes operaciones matemática, así las funciones trigonométricas se calculan usando las series de Maclaurin. Escribir un programa que lea un número real: x (-10 < x < 10), y calcule e imprima el coseno de x (cos(x)), utilizando 15 términos de la serie siguiente:

$$\cos(x) = 1 - \frac{x^2}{2!} + \frac{x^4}{4!} - \frac{x^6}{6!} + \cdots + \frac{(-1)^n x^{2n}}{(2n)!}$$

```
#include <iostream>
#include <math.h>
using namespace std;
main()
 {
    double x, cos=1, fact=1;
    cin>>x;
    for(int i=2; i<30; i=i+2){
       fact=-i*(i-1)*fact;
       cos=cos+pow(x,i)/double(fact);
    }
    cout<<cos<<endl;
 }
```

EJERCICIOS PROPUESTOS PARA RESOLVER EN SALA:

Problema7:
Escribir un programa que lea un número natural: n (0 < n < 100), e imprima la cantidad total de círculos que conforman un triángulo completo de lado n. Para los primeros 10 números naturales, la cantidad de círculos de los triángulos son: 1 3 6 10 15 21 28 36 45 55, como se muestra en la siguiente figura:



EJEMPLO DE ENTRADA
4
EJEMPLO DE SALIDA
10

Problema8:
Escribir un programa que lea una frase e imprima cada palabra en una línea separada
EJEMPLO DE ENTRADA
Esta es una frase de ejemplo
EJEMPLO DE SALIDA
Esta
es
una
frase
de
ejemplo

Problema9:
En el ingreso de datos por el teclado, es usual que los usuarios cometan errores de tipeado, provocando errores de datos y hasta el riesgo de que el programa colapse, por ejemplo, cuando se esperaba un número y por error se ingresan alguna letra o símbolo; para evitar esta situación, los datos ingresados deben ser validados y en el caso de que se detecte un error, el programa indica al usuario que cometió un error y que debe reingresar el número, con lo que se evitaría la interrupción del programa.
Escribir un programa que lea una cadena: string n, que corresponde a un número natural, verifique que todos los caracteres corresponden a dígitos numéricos, lo convierta en un número e imprima que corresponde a un número válido y su doble; en el caso que la cadena contenga un carácter que no sea numérico, imprima que no corresponde a un número válido.

EJEMPLO DE ENTRADA
234S8
EJEMPLO DE SALIDA
234S8, no es un número válido.

EJEMPLO DE ENTRADA
982353
EJEMPLO DE SALIDA
982353, es un número válido, su doble es: 1964706

Problema10:
Escribir un programa que lea un número natural: n (1 < n < 80), e imprima el borde de un rectángulo, formado por asteriscos, con una altura igual a n y una base igual a 2*n, como se muestra en el ejemplo siguiente:

EJEMPLO DE ENTRADA
5
EJEMPLO DE SALIDA
****
* *
* *
* *
****

EJERCICIOS PROPUESTOS PARA RESOLVER EXTRACLASE:

Problema11:
Escribir un programa que lea una cadena: string bin (con un tamaño máximo de 31 caracteres), el que corresponde a un número binario, el que puede tener ceros a la izquierda; a continuación, lo convierta en su equivalente en base 10 y lo imprima.

EJEMPLO DE ENTRADA
00101
EJEMPLO DE SALIDA
5

EJEMPLO DE ENTRADA
001110001
EJEMPLO DE SALIDA
113

Problema12:
El 6 de junio de cada año, la UAJMS cumple su aniversario y como parte de los festejos, en un solemne acto académico, se reconoce y premia a los estudiantes con los tres mejores promedios de cada carrera; para ello, vicerrectorado solicita a la DTIC proporcione esa información. Escribir un programa que lea una serie de nombres de estudiantes de una carrera específica y la nota promedio anual que les corresponda, hasta el ingreso de la palabra: "fin", que solo indica el final de la entrada de los datos; luego el programa imprimirá los nombres y notas de los tres estudiantes con los mejores promedios de esa carrera, en orden descendente de notas, como se muestra en el siguiente ejemplo (asuma que los nombres y las notas son todos diferentes):

EJEMPLO DE ENTRADA
Salvatierra María
67.52
Quiroga José Antonio
73.95
Fernández Pamela
71.96
Tapia Manuel
87.35
Santos Mónica
90.63
Méndez Fernando
62.83
Miranda Sergio
52.94
fin
EJEMPLO DE SALIDA

1: Santos Mónica 90.63
2: Tapia Manuel 87.35
3: Quiroga José Antonio 73.95

Problema13:
El procedimiento más simple para determinar si un número es primo o no, es contar el número de divisores; si un número n tiene dos divisores, entonces el número es primo. Por el contrario, en ocasiones, es útil conocer el número con la mayor cantidad de divisores. Escribir un programa que lea un número natural: n (1 < n < 1e6), e imprima, en una línea, el/los número/s entre 1 y n que tiene/n más divisores en ese rango, en orden ascendente, seguido por la cantidad de divisores que tiene/n.
EJEMPLO DE ENTRADA
20
EJEMPLO DE SALIDA
12 18 20; 6 divisores
EJEMPLO DE ENTRADA
400
EJEMPLO DE SALIDA
360; 24 divisores

Problema14:
Escribir un programa que lea un número natural: n (1 < n < 80), e imprima el borde de un rombo de lados igual a n, que está formado por asteriscos y tiene una disposición vertical, como se muestra en el ejemplo siguiente:
EJEMPLO DE ENTRADA
4
EJEMPLO DE SALIDA

```
   *
  * *
 *   *
*     *
 *   *
  * *
   *
```

Problema15:
Repetir el problema 4 de esta guía, para calcular el seno de un número x. La serie de Maclaurin para el seno es:

$$\sin(x) = x - \frac{x^3}{3!} + \frac{x^5}{5!} - \frac{x^7}{7!} + \cdots + \frac{(-1)^n x^{2n+1}}{(2n+1)!}$$

Problema16:
El factorial de un número natural: n, se define como: n! = 1*2*3*4*…*n; así por ejemplo:
0! = 1, por definición
1! = 1
2! = 1*2 = 2
3! = 1*2*3 = 6
4! = 1*2*3*4 = 24
….

Escribir un programa que lea un número natural: n (0 < n < 1e7), e imprima si n es el factorial de algún número o no lo es, como se muestra en los ejemplos siguientes.

EJEMPLO DE ENTRADA
6
EJEMPLO DE SALIDA
6 es le factorial de 3

EJEMPLO DE ENTRADA
20
EJEMPLO DE SALIDA
20 no es factorial de ningún número.