MATERIA: Programación I
GUIA DE LABORATORIO #: 14 y 15

COMPETENCIAS:

1. Describir las características de las estructuras repetitivas
2. Escribir programas usando la estructura repetitiva: for

TEORÍA ASOCIADA:

1. CARACTERÍSTICAS DE LAS ESTRUCTURAS REPETITIVAS

En programación, las estructuras repetitivas reflejan las acciones, rutinas o procesos repetitivos que el ser humano frecuentemente realiza o percibe en diversos aspectos de su vida diaria.
Una estructura repetitiva, es una sentencia que se caracteriza por ejecutar un mismo conjunto de instrucciones (bloque de código) cierta cantidad finita de veces; por este motivo la estructura repetitiva se denomina también estructura iterativa, ciclo o bucle (loop).

El control de repetición está formado por una condición que es una expresión lógica, la que, mientras sea verdadera, el bloque repetitivo se ejecuta reiteradamente y en el caso de dar falsa, entonces se finaliza la ejecución del bloque repetitivo y se continúa con la ejecución del resto de las instrucciones que siguen a esa estructura repetitiva.
El bloque repetitivo puede ser un bloque vació (sin instrucciones) aunque generalmente está conformado por una o varias instrucciones, incluyendo otras estructuras de control, de tipo secuencial, condicional y/o repetitivas.

En C++, las estructuras de control son: for (para), while (mientras) y do-while (hacer-mientras); en esta guía se estudiará la estructura for.
En C++ y los lenguajes de programación derivados de C, (a diferencia de otros lenguajes de programación), las tres estructuras de control son intercambiables, es decir que cualquier algoritmo repetitivo que use una de las estructuras, puede reescribirse con cualquiera de las otras dos estructuras; pese a ello, usualmente una de las estructuras es más conveniente de ser usada que las otras, según las características del algoritmo.

2. LA ESTRUCTURA REPETITIVA: FOR

La estructura repetitiva: for, generalmente se usa cuando se conoce de antemano el número de veces que el bloque de código se repetirá.

La sintaxis de la estructura: for, es:

```
for(inicialización, condición; modificador)
{
// Instrucciones del bloque de código
}
```

Donde:

Inicialización, define el valor inicial de la variable de control de la estructura repetitiva.
Condición, expresión lógica que si es verdadera el bloque repetitivo se ejecuta
Modificador, expresión que cambia el valor de la variable de control.

Estos elementos de la estructura for son prescindibles y podrían no usarse y aún así la estructura repetitiva podría funcionar correctamente.

Por ejemplo, para imprimir cinco veces la palabra: INFORMÁTICA, el código será:

```
for(int i=1; i<=5; i=i+1)
{
    cout<<"INFORMATICA"<<endl;
}
```

Un cuidado que debe tenerse en cuenta al implementar estructuras repetitivas, es evitar definir bucles infinitos, que no por error no finalizan la ejecución del bloque repetitivo. Por ejemplo:

```
for(int i=1; i<=5; i=i-1)
{
    cout<<"INFORMATICA"<<endl;
}
```

EJERCICIOS RESUELTOS:

Problema1:

Escribir un programa que lea un número natural: n (0 < n < 100), e imprima la tabla del n que se usa en los cuadernos escolares.

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

int main(void)
{
    int n;
    cin >>n;
    for (int i=1;i<=10;i++)
      cout<<n<<"*"<<i<<"="<<n*i<<endl;
    return 0;
}
```

Problema2:

Escribir un programa que lea un número natural: n, (0 < n < 10) e imprima todos los valores pares menores o iguales a n, en orden descendente hasta el 0.

```
#include <iostream>
using namespace std;
int main(void)
{
    int n;
    cin >> n;
    for (int i = n; i >= 0; i--)
        if (i % 2 == 0)
            cout << i << " " << endl;
    return 0;
}
```

Problema3:

Escribir un programa que lea un número natural: int n, (0 <= n <= 2e9), e imprima la cantidad de dígitos que tiene el mismo.
EJEMPLO DE ENTRADA
270
EJEMPLO DE SALIDA
3

```
#include <iostream>
using namespace std;
int main(void)
{
    int n, digitos = 0;
    cin >> n;
    if (n == 0)
        cout << 1 << endl;
    else
    {
        for (; n > 0;)
        {
            digitos++;
            n = n / 10;
        }
        cout << digitos << endl;
    }
    return 0;
}
```

Problema4:

Escribir un programa para un profesor, que lea un número natural: n (0 < n < 10), que corresponden a la cantidad de exámenes de un estudiante, luego lea n números enteros: nota (0 <= nota <= 100) e imprima el promedio, redondeado al entero más próximo.
EJEMPLO DE ENTRADA
3
80 80 81
EJEMPLO DE SALIDA
80

EJEMPLO DE ENTRADA
3
80 80 81
EJEMPLO DE SALIDA
80

```
#include <iostream>
#include <math.h>
using namespace std;
int main(void)
{
    int n, nota, suma=0;
    float promedio;
    cin >> n;
    for (int i = 1; i <= n; i++){
        cin >> nota;
        suma=suma+nota;
    }
    promedio=suma*1./n;
    cout<< promedio<<endl;
    cout << round(promedio) << endl;
    return 0;
}
```

EJERCICIOS PROPUESTOS PARA RESOLVER EN SALA:

Problema5
Escribir un programa que lea un número natural: n (0 < n < 100), que representa la cantidad de palabras a ingresar, luego lea n palabras e imprima cuántas palabras tienen una, dos y tres letras.
EJEMPLO DE ENTRADA
5
El árbol de la vida
EJEMPLO DE SALIDA
Una letra: 0
Dos letras: 3
Tres letras: 0

Problema6
Escribir un programa que lea un número natural: n (0 < n < 100), luego lea n números naturales: xi (0 <= xi < 1e6), e imprimir, en la primera línea la lista de los números primos ingresados, separados por un simple espacio; y en la segunda línea, la cantidad de números primos existentes.
EJEMPLO DE ENTRADA
5
27 31 5 12 2
EJEMPLO DE SALIDA
31 5 2
3

EJERCICIOS PROPUESTOS PARA RESOLVER EXTRACLASE:

Problema7:
La empresa de limpieza "Clean", contrata personal por horas, para que realizar trabajos de limpieza de casas, departamentos, habitaciones, etc., y cancelan un monto fijo por hora de 15 Bs/h. Escribir un programa que lea un número entero: n (0 < n < 10), que es el número de días que un empleado trabajó, y luego lea n valores naturales: horasi (0 < horasi <= 8), que representa la cantidad de horas que trabajó en un día; luego imprimir el monto de dinero que ese empleado debe cobrar.
EJEMPLO DE ENTRADA
3
4 2 7
EJEMPLO DE SALIDA
195

Problema8:
Una embolsadora de manzanas tiene una balanza donde van cayendo las manzanas desde un dispensador; las manzanas van embolsándose y apenas supere los 2 kg, un dispositivo de control detiene el dispensador de manzanas para luego el operario retire la bolsa y coloque una nueva bolsa para repetir el procedimiento.
Escribir un programa que lea una serie de valores que corresponden a los pesos de varias manzanas que van almacenándose en la bolsa y apenas se supere los 2 kg., imprima el mensaje "BOLSA LLENA", luego imprima el peso total de las manzanas y el número de manzanas, como se muestra en el ejemplo de salida siguiente:
EJEMPLO DE ENTRADA
0.12
0.09
0.14
0.11
0.1
0.14
0.18
0.12
0.15

0.16
0.13
0.11
0.09
0.13
0.15
0.1

EJEMPLO DE SALIDA
BOLSA LLENA
2.02 Kg
16 manzanas

Problema9:
Decide ahorrar algo de dinero y abre una cuenta de ahorro en el banco; cada inicio de mes deposita un monto de dinero y el banco le otorga un interés mensual de interés del 0.5% sobre el monto que tiene depositado durante un mes (sin incluir los intereses acumulados). Desea estimar el total de dinero que tendrá después de n meses y habiendo abonado cierta cantidad de dinero cada inicio de mes. Escribir un programa que lea un número natural: n (0 < n <=12) que representa la cantidad de meses que abonará dinero al banco, luego lea n números naturales: Mi, que representa la cantidad de dinero que abonará en un mes determinado, e imprima el total de dinero que debería cobrar después de los n meses.
EJEMPLO DE ENTRADA
2
500 100
EJEMPLO DE SALIDA
605.5

EJEMPLO DE ENTRADA
5
200 350 100 150 100
EJEMPLO DE SALIDA
915.5