**Instituto Superior de Formación Técnica Nº 151** logo151-trans.gif **Carrera: Analista de Sistemas  
1 Año. Algoritmos y Estructuras de Datos I**

|  |  |
| --- | --- |
| **Trabajo Práctico Nº 4** | **Unidad 4** |
| **Modalidad:** Semi-Presencial | **Estratégica Didáctica:** Trabajo Grupal. |
| **Metodología de Desarrollo:** acordar | **Metodología de Corrección:** acordar docente |
| **Carácter de Trabajo:** Obligatorio – con Nota | **Fecha Entrega:** A confirmar por el Docente. |

**Marco Teórico:**

1. ¿Describir el Concepto de Array?
2. Dar un Ejemplo de:
   1. Creación de Arrays (preferentemente con typeof)
   2. Asignación de Arrays
   3. Recorrido de Arrays.
3. Describir el Concepto de Listas (sin estructura)
   1. Dar un ejemplo de Inserción de un elemento
   2. Dar un ejemplo de Borrado de un elemento
4. Describir el Tipo de Dato “String”
   1. Dar ejemplos de 4 Funciones “Incluidas” en el Tipo “String”
   2. Que sentencia de Entrada usaría para leer un String con espacio x teclado
5. Describir la sentencia de repetición Do While
   1. Qué diferencia tiene con la sentencia While
   2. Dar 2 ejemplos de Uso de la Sentencia.

**Marco Practico:**

1. Realizar un programa Que rellene un array con los 100 primeros números enteros y los muestre en pantalla
2. Realizar un Programa Que rellene un array con los números primos comprendidos entre 1 y 100 y los muestre en pantalla
3. Realizar un Programa **Que rellene un array con los números impares comprendidos entre 1 y 100 y los muestre en pantalla**

Lic. Oemig José Luis.

1. Concepto de Array

Un array es una estructura de datos que almacena una colección de elementos del mismo tipo en una secuencia de posiciones contiguas en la memoria. Los arrays permiten un acceso rápido a cualquier elemento mediante un índice numérico.

2. Ejemplo de Creación, Asignación y Recorrido de Arrays

2.1 Creación de Arrays

int main()

{ int array[5]; // Creación de un array de 5 elementos de tipo int

return 0;

}

2.2 Asignación de Arrays

int main()

{ int array[5]; // Creación de un array de 5 elementos de tipo int

array[0] = 10;

array[1] = 20;

array[2] = 30;

array[3] = 40;

array[4] = 50;

return 0;

}

2.3 Recorrido de Arrays

int main()

{ int array[5] = {10, 20, 30, 40, 50};

for(int i = 0; i < 5; i++)

{ cout << "Elemento en la posición " << i << ": " << array[i] << endl;

}

return 0;

}

// En el material de la clase se monstruo el concepto de lista como un array el cual debemos llevar nosotros el indice mediante una variable por ejemplo int indicearray o int contador.

A continuacion se describe el concepto y utilizacion de Listas manejados por la biblioteca <list> de C++

3. Concepto de Listas

Una lista es una colección ordenada de elementos en la que se pueden insertar y eliminar elementos de manera eficiente. Las listas no requieren un tamaño predefinido y pueden crecer y reducirse dinámicamente.

3.1 Inserción de un Elemento

#include <iostream>

#include <list>

using namespace std;

int main()

{ list<int> lista;

lista.push\_back(10); // Inserta 10 al final de la lista

lista.push\_front(20); // Inserta 20 al principio de la lista

return 0;

}

3.2 Borrado de un Elemento

#include <iostream>

#include <list>

using namespace std;

int main()

{ list<int> lista = {10, 20, 30};

lista.pop\_back(); // Elimina el último elemento (30)

lista.pop\_front(); // Elimina el primer elemento (10)

return 0;

}

4. Tipo de Dato “String”

Un string es una secuencia de caracteres que se utiliza para representar texto. En C++, los strings pueden manejarse mediante arrays de caracteres terminados en nulo o mediante la clase string de la biblioteca estándar.

4.1 Ejemplos de 4 Funciones Incluidas en el Tipo “String”

#include <iostream>

#include <string>

using namespace std;

int main()

{ string str = "Hola Mundo";

cout << "Longitud: " << str.length() << endl; // Longitud de la cadena

cout << "Subcadena: " << str.substr(0, 4) << endl; // Subcadena

cout << "Posición de 'Mundo': " << str.find("Mundo") << endl; // Encontrar una subcadena

string str2 = "!";

cout << "Concatenación: " << str + str2 << endl; // Concatenación

return 0;

}

4.2 Sentencia de Entrada para Leer un String con Espacios desde el Teclado

#include <iostream>

#include <string>

using namespace std;

int main()

{ string str;

cout << "Introduce una frase: ";

getline(cin, str); // Lee una línea completa incluyendo espacios

cout << "Has introducido: " << str << endl;

return 0;

}

5. Sentencia de Repetición Do While

La sentencia do while ejecuta un bloque de código al menos una vez y luego repite la ejecución mientras una condición sea verdadera.

5.1 Diferencia con la Sentencia While

while: La condición se evalúa antes de ejecutar el bloque de código.

do while: El bloque de código se ejecuta al menos una vez antes de evaluar la condición.

5.2 Ejemplos de Uso de la Sentencia Do While

#include <iostream>

using namespace std;

int main()

{ int i = 1;

// Ejemplo 1: Imprimir números del 1 al 5

do {

cout << i << endl;

i++;

} while(i <= 5);

// Ejemplo 2: Leer hasta que se introduzca un número positivo

int num;

do {

cout << "Introduce un número positivo: ";

cin >> num;

} while(num <= 0);

return 0;

}