



ESCUELA POLITÉCNICA NACIONAL

ESCUELA DE FORMACIÓN DE TECNÓLOGOS



PROGRAMACION TDSD214

ASIGNATURA:

Programación

PROFESOR:

Ing. Ivonne Maldonado

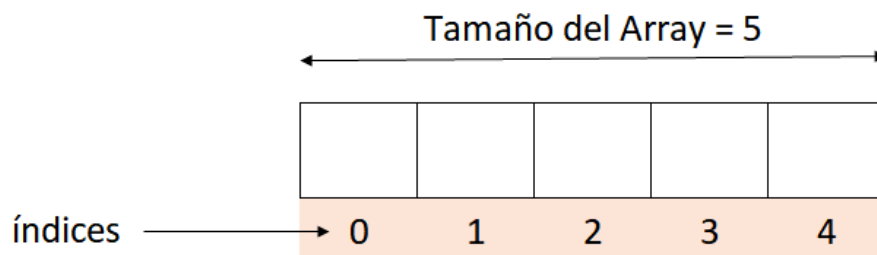
PERÍODO ACADÉMICO:

2023-B

DEBER 3

TÍTULO:

ARREGLOS



Arreglos en C++

Nombre:

Guerra Lovato Josué Eduard

PROPÓSITO DE LA PRÁCTICA

Familiarizar al estudiante con el uso de arreglos en el lenguaje C++.

OBJETIVO GENERAL

Utilizar arreglos en la solución de problemas, Buscando una comprensión sólida de los conceptos fundamentales relacionados con arreglos, incluyendo, pero no limitado a índices, recorridos, manipulación de elementos y dimensiones.

OBJETIVO ESPECÍFICOS

- Manipulación de arreglos multidimensionales.
- Manipulación avanzada de elementos.
- Dominio de índices y recorridos.

TAREA

- Solicite números al usuario y almacene los pares en un arreglo y los impares en otro arreglo.

```
1  #include <iostream>
2  #include <vector>
3
4  using namespace std;
5
6  int main() {
7
8      cout << "    ---// Bienvenido //--    " << endl;
9      cout << " Pares e impares en arreglos " << endl;
10     cout << " Inserte su nombre para comenzar: " << endl;
11     string nombre;
12     cin >> nombre;
13     const int MAX_NUMEROS = 100;
14     int pares[MAX_NUMEROS];
15     int impares[MAX_NUMEROS];
16
17     int n;
18     cout << "Ingrese la cantidad de numeros: ";
19     cin >> n;
20
21     int indicePares = 0;
22     int indiceImpares = 0;
23
24     for (int i = 0; i < n; ++i) {
25         int numero;
26         cout << "Ingrese el numero " << i + 1 << ": ";
27         cin >> numero;
28
29         if (numero % 2 == 0) {
30             pares[indicePares] = numero;
31             indicePares++;
32         } else {
33             impares[indiceImpares] = numero;
34             indiceImpares++;
35         }
36     }
37
38     cout << "Numeros pares: ";
39     cout << "[ ";
40     for (int i = 0; i < indicePares; ++i) {
41         cout << pares[i] << " - ";
42     }
43     cout << " ]" << endl;
44     cout << endl;
45
46     cout << "Numeros impares: ";
47     cout << "[ ";
48     for (int i = 0; i < indiceImpares; ++i) {
49         cout << impares[i] << " ";
50     }
51     cout << " ]" << endl;
52     cout << endl;
53     return 0;
54 }
```

Ejecución:

```
E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio1_Par-Impares_GuerraJosue.exe
---// Bienvenido //--
Pares e impares en arreglos
Inserte su nombre para comenzar:
Josue
Ingrese la cantidad de numeros: 10
Ingrese el numero 1: 5
Ingrese el numero 2: 2
Ingrese el numero 3: 1
Ingrese el numero 4: 6
Ingrese el numero 5: 8
Ingrese el numero 6: 9
Ingrese el numero 7: 7
Ingrese el numero 8: 45
Ingrese el numero 9: 3
Ingrese el numero 10: 1
Numeros pares: [ 2 - 6 - 8 - ]

Numeros impares: [ 5 1 9 7 45 3 1 ]

-----
Process exited after 56.99 seconds with return value 0
Presione una tecla para continuar . . . █
```

El programa como tal solicita al usuario que inserte la cantidad del arreglo y con eso también los valores del mismo de ahí según el número insertado este se retornará como dentro de otro arreglo el cual detecta si es par o impar y así los va ordenando contando desde la posición 0 para acomodar en cada lugar correspondiente, como tal se tiene en cuenta que el código también realiza la operación respectiva para ver si su módulo es o no par, el inicio donde solicita el nombre solo es como parte del programa para darle un poco de formalidad y explicar de manera corta al usuario lo que realizará dentro del programa presentado.

EL EJERCICIO 2 SE ENCUENTRA EN LA PÁGINA SIGUIENTE:

- Solicite números al usuario y almacene en un arreglo solo los divisibles para 3.

```

1  #include <iostream>
2  #include <vector>
3
4  using namespace std;
5
6  int main() {
7
8      cout << "    ---// Bienvenido //--    " << endl;
9      cout << "Arreglo son con divisibles para 3 " << endl;
10     cout << "Inserte su nombre para comenzar: " << endl;
11     string nombre;
12     cin >> nombre;
13
14     const int numero = 100;
15     int divisiblesPor3[numero];
16
17     int n;
18     cout << "Ingrese la cantidad de numeros: ";
19     cin >> n;
20
21     int divisibles = 0;
22
23     for (int i = 0; i < n; ++i) {
24         int numero;
25         cout << "Ingrese el numero " << i + 1 << ": ";
26         cin >> numero;
27
28         if (numero % 3 == 0) {
29
30             divisiblesPor3[divisibles] = numero;
31             divisibles++;
32         }
33     }
34
35     cout << "Numeros divisibles por 3: ";
36     for (int i = 0; i < divisibles; ++i) {
37         cout << divisiblesPor3[i] << " ";
38     }
39     cout << endl;
40
41     return 0;
42 }

```

Ejecución:

```

E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio2_A-Divisiblespara3_GuerraJosue.exe
---// Bienvenido //--
Arreglo son con divisibles para 3
Inserte su nombre para comenzar:
Josue
Ingrese la cantidad de numeros: 8
Ingrese el numero 1: 1
Ingrese el numero 2: 2
Ingrese el numero 3: 8
Ingrese el numero 4: 6
Ingrese el numero 5: 4
Ingrese el numero 6: 9
Ingrese el numero 7: 3
Ingrese el numero 8: 7
Numeros divisibles por 3: 6 9 3

-----
Process exited after 57.62 seconds with return value 0
Presione una tecla para continuar . . .

```

El programa como tal realizara el calculo correspondiente para saber si el número ingresado es divisible para 3, con eso los ubica dentro de un arreglo y muestra en pantalla dichos números, tomando en cuenta que también se solicita al usuario la cantidad de números que desea ingresar y los mismos cumplen con lo mencionado anteriormente.

- Sume dos matrices

```
1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int filas, columnas;
6      cout << "    ---// Bienvenido //--    " << endl;
7      cout << "    Suma de dos matrices    " << endl;
8      cout << "    Inserte su nombre para comenzar: " << endl;
9      string nombre;
10     cin >> nombre;
11
12     cout << "Ingrese el numero de filas: ";
13     cin >> filas;
14     cout << "Ingrese el numero de columnas: ";
15     cin >> columnas;
16
17     int matriz1[filas][columnas];
18     int matriz2[filas][columnas];
19
20     cout << "Ingrese los elementos de la primera matriz:" << endl;
21     for (int i = 0; i < filas; ++i) {
22         for (int j = 0; j < columnas; ++j) {
23             cout << "Matriz1[" << i << "][" << j << "]: ";
24             cin >> matriz1[i][j];
25         }
26     }
27
28     cout << "Ingrese los elementos de la segunda matriz:" << endl;
29     for (int i = 0; i < filas; ++i) {
30         for (int j = 0; j < columnas; ++j) {
31             cout << "Matriz2[" << i << "][" << j << "]: ";
32             cin >> matriz2[i][j];
33         }
34     }
35
36     cout << "\nLa Matriz 1 es:" << endl;
37     for (int i = 0; i < filas; ++i) {
38         for (int j = 0; j < columnas; ++j) {
39             cout << matriz1[i][j] << "\t";
40         }
41         cout << endl;
42     }
43
44     cout << "\n LaMatriz 2 es:" << endl;
45     for (int i = 0; i < filas; ++i) {
46         for (int j = 0; j < columnas; ++j) {
47             cout << matriz2[i][j] << "\t";
48         }
49         cout << endl;
50     }
51
52     int resultado[filas][columnas];
53     for (int i = 0; i < filas; ++i) {
54         for (int j = 0; j < columnas; ++j) {
55             resultado[i][j] = matriz1[i][j] + matriz2[i][j];
56         }
57     }
58
59     cout << "\nla matriz resultado es:" << endl;
60     for (int i = 0; i < filas; ++i) {
61         for (int j = 0; j < columnas; ++j) {
62             cout << resultado[i][j] << "\t";
63         }
64         cout << endl;
65     }
66
67     return 0;
68 }
```

La ejecución se encuentra en la siguiente página

Ejecución:

```
E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio3_SumaMatrices_GuerraJosue.exe
---// Bienvenido //--
Suma de dos matrices
Inserte su nombre para comenzar:
Josue
Ingrese el numero de filas: 2
Ingrese el numero de columnas: 3
Ingrese los elementos de la primera matriz:
Matriz1[0][0]: 1
Matriz1[0][1]: 5
Matriz1[0][2]: 9
Matriz1[1][0]: 6
Matriz1[1][1]: 2
Matriz1[1][2]: 8
Ingrese los elementos de la segunda matriz:
Matriz2[0][0]: 7
Matriz2[0][1]: 5
Matriz2[0][2]: 3
Matriz2[1][0]: 2
Matriz2[1][1]: 1
Matriz2[1][2]: 5

La Matriz 1 es:
1      5      9
6      2      8

LaMatriz 2 es:
7      5      3
2      1      5

La matriz resultado es:
8      10     12
8      3      13

-----
Process exited after 20.78 seconds with return value 0
Presione una tecla para continuar . . .
```

Inicialmente al usuario se le solicita el número de filas y columnas que desea para realizar la operación indicada dentro del problema tomando en cuenta que ambas matrices deben ser de la misma magnitud ya que así se podrán sumar correctamente, y al referido en magnitud que sean iguales ya que igual pueden tener diferentes números de filas y columnas pero para ambas matrices del mismo tamaño, después de eso se compara y se van sumando los elementos ubicados en la posición de cada matriz y al final mostrara la suma de las mismas.

EL EJERCICIO 4 SE ENCUENTRA EN LA SIGUIENTE PÁGINA

- Determine si una matriz es simétrica

```

1  #include <iostream>
2
3  using namespace std;
4
5  int main() {
6      cout << "    ---// Bienvenido //--    " << endl;
7      cout << "    Verificar simetria de una matriz " << endl;
8      cout << "    Inserte su nombre para comenzar: " << endl;
9      string nombre;
10     cin >> nombre;
11     int filas, columnas;
12     do {
13         cout << "Ingrese el numero de filas: ";
14         cin >> filas;
15         cout << "Ingrese el numero de columnas: ";
16         cin >> columnas;
17
18         if (filas == columnas) {
19             int matriz[filas][columnas];
20             cout << "\nLa matriz ingresada por " << nombre << " es simetrica." << endl;
21             cout << "Ingrese los elementos de la matriz:" << endl;
22
23             for (int i = 0; i < filas; ++i) {
24                 for (int j = 0; j < columnas; ++j) {
25                     cout << "Matriz[" << i << "][" << j << "]: ";
26                     cin >> matriz[i][j];
27                 }
28             }
29             cout << "\nLa matriz es:" << endl;
30             for (int i = 0; i < filas; ++i) {
31                 cout << "[";
32                 for (int j = 0; j < columnas; ++j) {
33                     cout << " " << matriz[i][j] << " ";
34                     if (j < columnas - 1) {
35                         cout << " ";
36                     }
37                 }
38                 cout << "]" << endl;
39             }
40         } else {
41             cout << "\nLa matriz ingresada por " << nombre << " no es simetrica." << endl;
42         }
43     } while (filas != columnas);
44     return 0;
45 }

```

Ejecución:

```

E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio4_MatrizSimetrica_GuerraJosue.exe
    ---// Bienvenido //--
    Verificar simetria de una matriz
    Inserte su nombre para comenzar:
Josue
Ingrese el numero de filas: 1
Ingrese el numero de columnas: 6

La matriz ingresada por Josue no es simetrica.
Ingrese el numero de filas: 2
Ingrese el numero de columnas: 2

La matriz ingresada por Josue es simetrica.
Ingrese los elementos de la matriz:
Matriz[0][0]: 1
Matriz[0][1]: 5
Matriz[1][0]: 9
Matriz[1][1]: 6

La matriz es:
[ 1  5 ]
[ 9  6 ]

-----
Process exited after 11.17 seconds with return value 0
Presione una tecla para continuar . . .

```

- **Multiplique dos matrices**

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main() {
6      int filas, columnas;
7      cout << "      ---// Bienvenido //--      " << endl;
8      cout << "      Suma de dos matrices " << endl;
9      cout << " Inserte su nombre para comenzar: " << endl;
10     string nombre;
11     cin >> nombre;
12
13     cout << "Ingrese el numero de filas: ";
14     cin >> filas;
15     cout << "Ingrese el numero de columnas: ";
16     cin >> columnas;
17
18     int matriz1[filas][columnas];
19     int matriz2[filas][columnas];
20
21     cout << "Ingrese los elementos de la primera matriz:" << endl;
22     for (int i = 0; i < filas; ++i) {
23         for (int j = 0; j < columnas; ++j) {
24             cout << "Matriz1[" << i << "][" << j << "]: ";
25             cin >> matriz1[i][j];
26         }
27     }
28
29     cout << "Ingrese los elementos de la segunda matriz:" << endl;
30     for (int i = 0; i < filas; ++i) {
31         for (int j = 0; j < columnas; ++j) {
32             cout << "Matriz2[" << i << "][" << j << "]: ";
33             cin >> matriz2[i][j];
34         }
35     }
36
37     cout << "\nLa Matriz 1 es:" << endl;
38     for (int i = 0; i < filas; ++i) {
39         for (int j = 0; j < columnas; ++j) {
40             cout << matriz1[i][j] << "\t";
41         }
42         cout << endl;
43     }
44
45     cout << "\n LaMatriz 2 es:" << endl;
46     for (int i = 0; i < filas; ++i) {
47         for (int j = 0; j < columnas; ++j) {
48             cout << matriz2[i][j] << "\t";
49         }
50         cout << endl;
51     }
52
53     int resultado[filas][columnas];
54     for (int i = 0; i < filas; ++i) {
55         for (int j = 0; j < columnas; ++j) {
56             resultado[i][j] = matriz1[i][j] * matriz2[i][j];
57         }
58     }
59
60     cout << "\nLa matriz resultado es:" << endl;
61     for (int i = 0; i < filas; ++i) {
62         for (int j = 0; j < columnas; ++j) {
63             cout << resultado[i][j] << "\t";
64         }
65         cout << endl;
66     }
67
68     return 0;
69 }
```

El código toma la misma sintaxis que el ejercicio 3 ya que aplica los mismos parámetros lo único que hace es cambiar el símbolo de la operación y al final muestra la matriz con la operación indicada.

La ejecución se encuentra en la siguiente página.

Ejecución:

```
E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio5_MultiplicacionMatrices_GuerraJosue.exe

---// Bienvenido //--
Suma de dos matrices
Inserte su nombre para comenzar:
EDUARD
Ingrese el numero de filas: 2
Ingrese el numero de columnas: 2
Ingrese los elementos de la primera matriz:
Matriz1[0][0]: 1
Matriz1[0][1]: 5
Matriz1[1][0]: 6
Matriz1[1][1]: 8
Ingrese los elementos de la segunda matriz:
Matriz2[0][0]: 3
Matriz2[0][1]: 5
Matriz2[1][0]: 4
Matriz2[1][1]: 9

La Matriz 1 es:
1      5
6      8

LaMatriz 2 es:
3      5
4      9

La matriz resultado es:
3      25
24     72

-----
Process exited after 15.49 seconds with return value 0
Presione una tecla para continuar . . .
```

El código toma la misma sintaxis que el ejercicio 3 ya que aplica los mismos parámetros lo único que hace es cambiar el símbolo de la operación y al final muestra la matriz con la operación indicada.

EL EJERCICIO 6 SE ENCUENTRA DESPUÉS DE ESTE SALTO DE PÁGINA.

- Genera e imprime una matriz identidad (una matriz cuadrada con unos en la diagonal principal y ceros en el resto).

```
1  #include <iostream>
2
3  using namespace std;
4
5  int main() {
6      cout << "    ---// Bienvenido //--    " << endl;
7      cout << "    Matriz Identidad    " << endl;
8      cout << " Inserte su nombre para comenzar: " << endl;
9      string nombre;
10     cin >> nombre;
11
12     int dimension;
13     cout << "Ingrese la dimension de la matriz : ";
14     cin >> dimension;
15
16     if (dimension <= 0) {
17         cout << "La dimensión debe ser un número entero mayor que 0." << endl;
18         return 1;
19     }
20
21     cout << "La matriz identidad de dimension " << dimension << " es:" << endl;
22
23     for (int i = 0; i < dimension; ++i) {
24         for (int j = 0; j < dimension; ++j) {
25             if (i == j) {
26                 cout << " 1 ";
27             } else {
28                 cout << " 0 ";
29             }
30
31             if (j < dimension - 1) {
32                 cout << " ";
33             }
34         }
35         cout << endl;
36     }
37
38     return 0;
39 }
```

Ejecución:

```
E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio6_MatrizIdentidad_GuerraJosue.exe
    ---// Bienvenido //--
    Matriz Identidad
Inserte su nombre para comenzar:
Eduard
Ingrese la dimension de la matriz : 4
La matriz identidad de dimension 4 es:
1  0  0  0
0  1  0  0
0  0  1  0
0  0  0  1

-----
Process exited after 12.47 seconds with return value 0
Presione una tecla para continuar . . .
```

- Suma los elementos de cada fila y cada columna en una matriz y muestre los resultados.

```

1  #include <iostream>
2  using namespace std;
3
4  void mostrarMatriz(int** matriz, int filas, int columnas) {
5      cout << "Matriz ingresada:" << endl;
6      for (int i = 0; i < filas; ++i) {
7          for (int j = 0; j < columnas; ++j) {
8              cout << matriz[i][j] << "\t";
9          }
10         cout << endl;
11     }
12 }
13 void mostrarSumaFilaColumna(int** matriz, int filas, int columnas) {
14     cout << "Suma de elementos por fila:" << endl;
15     for (int i = 0; i < filas; ++i) {
16         int sumaFila = 0;
17         for (int j = 0; j < columnas; ++j) {
18             sumaFila += matriz[i][j];
19         }
20         cout << "Fila " << i + 1 << ": " << sumaFila << endl;
21     }
22     cout << "Suma de elementos por columna:" << endl;
23     for (int j = 0; j < columnas; ++j) {
24         int sumaColumna = 0;
25         for (int i = 0; i < filas; ++i) {
26             sumaColumna += matriz[i][j];
27         }
28         cout << "Columna " << j + 1 << ": " << sumaColumna << endl;
29     }
30     int sumaTotal = 0;
31     for (int i = 0; i < filas; ++i) {
32         for (int j = 0; j < columnas; ++j) {
33             sumaTotal += matriz[i][j];
34         }
35     }
36     cout << "Suma total de todos los elementos: " << sumaTotal << endl;
37 }
38
39 int main() {
40     int filas, columnas;
41
42     cout << "Ingrese el número de filas: ";
43     cin >> filas;
44     cout << "Ingrese el número de columnas: ";
45     cin >> columnas;
46
47     int** matriz = new int*[filas];
48     for (int i = 0; i < filas; ++i) {
49         matriz[i] = new int[columnas];
50     }
51
52     cout << "Ingrese los elementos de la primera matriz:" << endl;
53     for (int i = 0; i < filas; ++i) {
54         for (int j = 0; j < columnas; ++j) {
55             cout << "Matriz[" << i << "][" << j << "]: ";
56             cin >> matriz[i][j];
57         }
58     }
59     mostrarMatriz(matriz, filas, columnas);
60     mostrarSumaFilaColumna(matriz, filas, columnas);
61     return 0;
62 }

```

Ejecución:

```

E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio7_SumaEnMatrices_GuerraJosue.exe
Ingrese el n-mero de filas: 2
Ingrese el n-mero de columnas: 4
Ingrese los elementos de la primera matriz:
Matriz[0][0]: 1
Matriz[0][1]: 5
Matriz[0][2]: 9
Matriz[0][3]: 6
Matriz[1][0]: 8
Matriz[1][1]: 7
Matriz[1][2]: 4
Matriz[1][3]: 3
Matriz ingresada:
1      5      9      6
8      7      4      3
Suma de elementos por fila:
Fila 1: 21
Fila 2: 22
Suma de elementos por columna:
Columna 1: 9
Columna 2: 12
Columna 3: 13
Columna 4: 9
Suma total de todos los elementos: 43

-----
Process exited after 10.97 seconds with return value 0
Presione una tecla para continuar . . .

```

- Ordena de manera ascendente un vector utilizando el algoritmo de ordenamiento burbuja.

```

1  #include <iostream>
2  using namespace std;
3  int main() {
4      int n;
5
6      cout << "    ---// Bienvenido //--    " << endl;
7      cout << "    Orden de vectores " << endl;
8      cout << "    Inserte su nombre para comenzar: " << endl;
9      string nombre;
10     cin >> nombre;
11
12     cout << "Ingrese la longitud del vector: ";
13     cin >> n;
14
15     int vector[n];
16     cout << "Ingrese los elementos del vector:" << endl;
17     for (int i = 0; i < n; ++i) {
18         cout << "Elemento " << i + 1 << ": ";
19         cin >> vector[i];
20     }
21
22     cout << "\nVector ingresado inicialmente:" << endl;
23     cout << "[ ";
24     for (int i = 0; i < n; ++i) {
25         cout << vector[i];
26         if (i < n - 1) {
27             cout << " - ";
28         }
29     }
30     cout << "]" << endl;
31     for (int i = 0; i < n - 1; ++i) {
32         for (int j = 0; j < n - i - 1; ++j) {
33             if (vector[j] > vector[j + 1]) {
34                 int temp = vector[j];
35                 vector[j] = vector[j + 1];
36                 vector[j + 1] = temp;
37             }
38         }
39     }
40
41     cout << "\nVector ordenado en orden ascendente:" << endl;
42     cout << "[ ";
43     for (int i = 0; i < n; ++i) {
44         cout << vector[i];
45         if (i < n - 1) {
46             cout << " - ";
47         }
48     }
49     cout << "]" << endl;
50
51     return 0;
52 }

```

Ejecución:

```

E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio7_SumaEnMatrices_GuerraJosue.exe
    ---// Bienvenido //--
    Orden de vectores
    Inserte su nombre para comenzar:
Josue-Eduard
Ingrese la longitud del vector: 6
Ingrese los elementos del vector:
Elemento 1: 1
Elemento 2: 5
Elemento 3: 9
Elemento 4: 6
Elemento 5: 3
Elemento 6: 2

Vector ingresado inicialmente:
[ 1 - 5 - 9 - 6 - 3 - 2 ]

Vector ordenado en orden ascendente:
[ 1 - 2 - 3 - 5 - 6 - 9 ]

-----
Process exited after 14.34 seconds with return value 0
Presione una tecla para continuar . . .

```

- Elimina los elementos duplicados de un vector y muestre el vector resultante.
- Este ejercicio se realizo de dos maneras diferentes.
- 1 usando funciones para ordenar.

```

1  #include <iostream>
2  #include <algorithm>
3
4  using namespace std;
5
6  int main() {
7      int n;
8
9      cout << "    ---// Bienvenido //--    " << endl;
10     cout << "        Eliminación de duplicados " << endl;
11     cout << "    Inserte su nombre para comenzar: " << endl;
12     string nombre;
13     cin >> nombre;
14     cout << "Ingrese la longitud del vector: ";
15     cin >> n;
16
17     int vector[n];
18     cout << "Ingrese los elementos del vector:" << endl;
19     for (int i = 0; i < n; ++i) {
20         cout << "Elemento " << i + 1 << ": ";
21         cin >> vector[i];
22     }
23
24     cout << "\nVector ingresado inicialmente:" << endl;
25     cout << "[ ";
26     for (int i = 0; i < n; ++i) {
27         cout << vector[i];
28         if (i < n - 1) {
29             cout << " , ";
30         }
31     }
32     cout << " ]" << endl;
33
34     sort(vector, vector + n);
35
36     int nuevoTamano = 1;
37     for (int i = 1; i < n; ++i) {
38         if (vector[i] != vector[i - 1]) {
39             vector[nuevoTamano++] = vector[i];
40         }
41     }
42
43     cout << "\nVector sin elementos duplicados:" << endl;
44     cout << "[ ";
45     for (int i = 0; i < nuevoTamano; ++i) {
46         cout << vector[i];
47         if (i < nuevoTamano - 1) {
48             cout << " , ";
49         }
50     }
51     cout << " ]" << endl;
52
53     return 0;
54 }

```

Ejecución:

```

E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio9_EliminarDuplicados-UsandoFunciones_GuerraJosue.exe
    ---// Bienvenido //--
        Eliminaci% de duplicados
    Inserte su nombre para comenzar:
Eduard
Ingrese la longitud del vector: 12
Ingrese los elementos del vector:
Elemento 1: 1
Elemento 2: 5
Elemento 3: 2
Elemento 4: 15
Elemento 5: 2
Elemento 6: 3
Elemento 7: 6
Elemento 8: 5
Elemento 9: 8
Elemento 10: 9
Elemento 11: 15
Elemento 12: 6

Vector ingresado inicialmente:
[ 1 , 5 , 2 , 15 , 2 , 3 , 6 , 5 , 8 , 9 , 15 , 6 ]

Vector sin elementos duplicados:
[ 1 , 2 , 3 , 5 , 6 , 8 , 9 , 15 ]

-----
Process exited after 25.37 seconds with return value 0
Presione una tecla para continuar . . .

```

- 2 Código normal

```
1  #include <iostream>
2  #include <algorithm>
3
4  using namespace std;
5
6  int main() {
7      int n;
8
9      cout << "    ---// Bienvenido //--    " << endl;
10     cout << "    Eliminación de duplicados " << endl;
11     cout << "    Inserte su nombre para comenzar: " << endl;
12     string nombre;
13     cin >> nombre;
14
15     cout << "Ingrese la longitud del vector: ";
16     cin >> n;
17
18     int vector[n];
19
20     cout << "Ingrese los elementos del vector:" << endl;
21     for (int i = 0; i < n; ++i) {
22         cout << "Elemento " << i + 1 << ": ";
23         cin >> vector[i];
24     }
25
26     cout << "\nVector ingresado inicialmente:" << endl;
27     cout << "[" << " ";
28     for (int i = 0; i < n; ++i) {
29         cout << vector[i];
30         if (i < n - 1) {
31             cout << " - ";
32         }
33     }
34     cout << "]" << endl;
35
36     sort(vector, vector + n);
37
38     int nuevoTamano = 1;
39     for (int i = 1; i < n; ++i) {
40         if (vector[i] != vector[i - 1]) {
41             vector[nuevoTamano++] = vector[i];
42         }
43     }
44
45     cout << "\nVector sin elementos duplicados:" << endl;
46     cout << "[" << " ";
47     for (int i = 0; i < nuevoTamano; ++i) {
48         cout << vector[i];
49         if (i < nuevoTamano - 1) {
50             cout << " - ";
51         }
52     }
53     cout << "]" << endl;
54
55     return 0;
56 }
```

Ejecución:

```
E:\Descargas\Deber 3\Ejercicios deber\#Ejercicio9_EliminarDuplicados_GuerraJosue.exe
Eliminaci% de duplicados
Inserte su nombre para comenzar:
Edjos
Ingrese la longitud del vector: 15
Ingrese los elementos del vector:
Elemento 1: 1
Elemento 2: 5
Elemento 3: 9
Elemento 4: 6
Elemento 5: 3
Elemento 6: 2
Elemento 7: 8
Elemento 8: 5
Elemento 9: 2
Elemento 10: 4
Elemento 11: 6
Elemento 12: 7
Elemento 13: 1
Elemento 14: 3
Elemento 15: 2

Vector ingresado inicialmente:
[ 1 - 5 - 9 - 6 - 3 - 2 - 8 - 5 - 2 - 4 - 6 - 7 - 1 - 3 - 2 ]

Vector sin elementos duplicados:
[ 1 - 2 - 3 - 4 - 5 - 6 - 7 - 8 - 9 ]

-----
Process exited after 12.75 seconds with return value 0
Presione una tecla para continuar . . .
```

CONSULTA

- Consulte como se retorna un arreglo en una función. (Recuerde esto se debe hacer con punteros 🏹)

RETORNO DE ARREGLOS EN UNA FUNCIÓN

Dentro de c++ no se puede retornar un arreglo directamente de una función, tomando que cuenta que los punteros no pueden llegar a ser copiados del todo, pero se lo puede llegar a devolver mediante un puntero que apunte hacia el primer elemento del arreglo.

En consideración cuando se trabaja con arreglos se trata con punteros, el cual se lo puede llegar a entender como bloque contiguo de memoria como tal siendo una dirección que apunta en el primer arreglo, este procede a apuntar la dirección en memoria.

Las formaciones en c++ son un tipo de datos con un comportamiento particular, se les aplica una regla especial conocida como decaimiento a puntero, que en términos generales viene a decir que si pasas una formación a una función ésta se transformará en puntero, En C/C++, cuando utilizamos la notación array[] como parámetro de una función, es importante comprender que en realidad se trata de un puntero al primer elemento del array proporcionado. En consecuencia, el prototipo de la función debe estar diseñado para devolver un puntero al tipo de dato almacenado en el array, que en este caso es int. Una vez que la función ha completado su ejecución, podemos acceder a los elementos del array mediante la notación de corchetes [] o desreferenciando directamente el puntero resultante.

Este enfoque es esencial para entender cómo se gestionan y manipulan los arrays en funciones, ya que permite trabajar con punteros y acceder a los datos del array de manera eficiente, facilitando así la manipulación y procesamiento de los elementos contenidos en él [1], [2].

- Ponga un ejemplo práctico de retorno de un arreglo.

```
[*] #Ejercicio1Consulta_Ejemplo_Guerralosue.cpp [*] #Ejercicio2Consulta_Ejemplo_Guerralosue.cpp
1 //Ejercicio de ejemplo
2 //retorno de arreglos
3 #include <iostream>
4 using namespace std;
5
6 // Función que retorna un puntero a un arreglo
7 int* crearArreglo(int tamano) {
8     // Se crea un arreglo estático en la función
9     static int arreglo[100]; // Se elige un tamaño máximo arbitrario
10
11     for (int i = 0; i < tamano; ++i) {
12         arreglo[i] = i * 3;
13     }
14     // Se retorna un puntero al primer elemento del arreglo
15     return arreglo;
16 }
17
18 int main() {
19     cout << "Ejemplo de retorno de arreglo con funciones en C++" << endl;
20
21     int tamano;
22     cout << "Ingrese el tamaño del arreglo: ";
23     cin >> tamano;
24
25     // Llama a la función y obtén el puntero al arreglo resultante
26     int* punteroArreglo = crearArreglo(tamano);
27
28     cout << "El arreglo retornado es: ";
29     for (int i = 0; i < tamano; ++i) {
30         cout << punteroArreglo[i] << " ";
31     }
32
33     return 0;
34 }
```

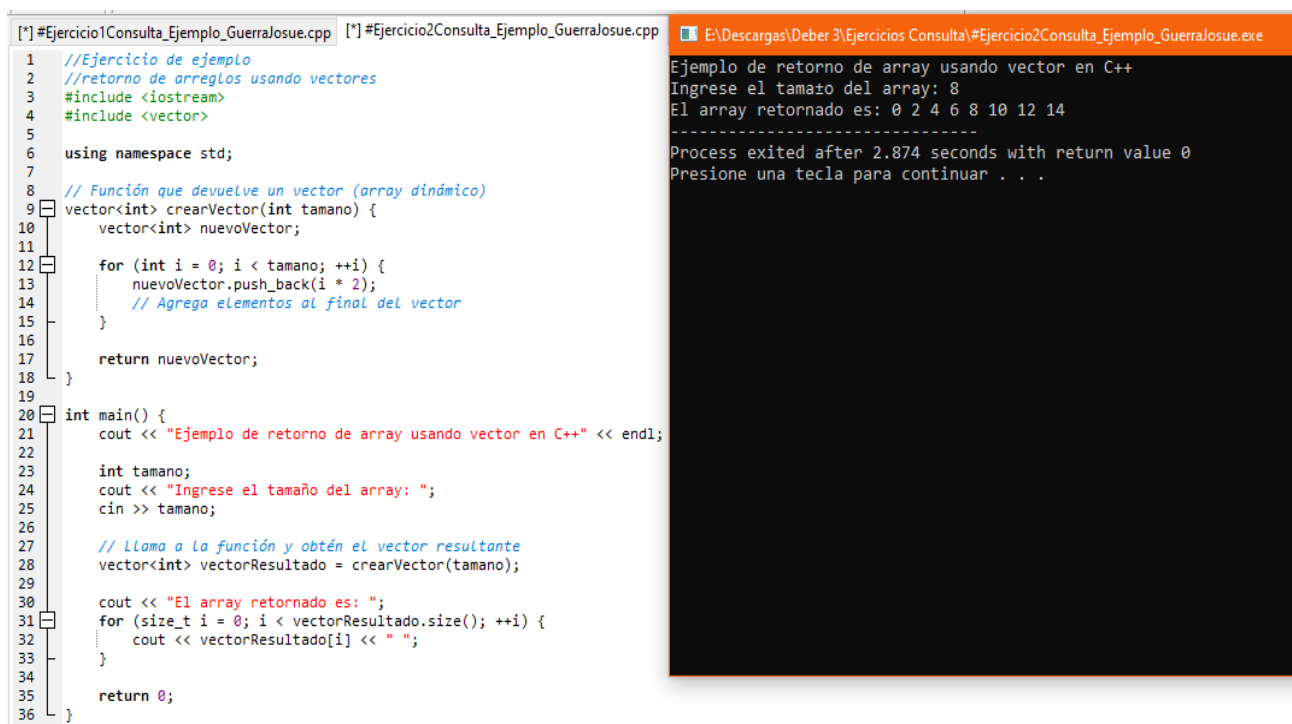
```
E:\Descargas\Deber 3\Ejercicios Consulta\Ejercicio1Consulta_Ejemplo_Guerralosue.exe
Ejemplo de retorno de arreglo con funciones en C++
Ingrese el tamaño del arreglo: 4
El arreglo retornado es: 0 3 6 9
-----
Process exited after 4.753 seconds with return value 0
Presione una tecla para continuar . . .
```

USAR EL VECTOR PARA DEVOLVER EL ARRAY EN LA FUNCIÓN

En C++, el contenedor vector es una estructura de datos dinámica que permite almacenar elementos de manera dinámica y flexible. Al utilizar vector para devolver un array desde una función, se evitan problemas asociados con la devolución de punteros a arreglos locales, mitigando así riesgos de comportamiento indefinido. vector proporciona una solución segura y eficiente, ya que gestiona dinámicamente la memoria y se adapta a la cantidad de elementos necesarios, facilitando el manejo de datos de manera robusta y evitando problemas comunes relacionados con la gestión manual de memoria.

Esto simplifica la gestión de memoria, mejora la seguridad y ofrece funcionalidades adicionales, como la capacidad de conocer el tamaño del vector dinámicamente. Al emplear vector, se promueve un código más moderno, mantenible y seguro, permitiendo a los desarrolladores concentrarse en la lógica del problema en lugar de preocuparse por detalles de gestión de memoria. Es crucial explorar y practicar activamente el uso de vector para aprovechar sus ventajas en términos de eficiencia y seguridad en la programación en C++ [2].

- Ponga un ejemplo práctico de retorno de un arreglo.



```
[*] #Ejercicio1Consulta_Ejemplo_Guerrajosue.cpp [*] #Ejercicio2Consulta_Ejemplo_Guerrajosue.cpp
1 //Ejercicio de ejemplo
2 //retorno de arreglos usando vectores
3 #include <iostream>
4 #include <vector>
5
6 using namespace std;
7
8 // Función que devuelve un vector (array dinámico)
9 vector<int> crearVector(int tamaño) {
10     vector<int> nuevoVector;
11
12     for (int i = 0; i < tamaño; ++i) {
13         nuevoVector.push_back(i * 2);
14         // Agrega elementos al final del vector
15     }
16
17     return nuevoVector;
18 }
19
20 int main() {
21     cout << "Ejemplo de retorno de array usando vector en C++" << endl;
22
23     int tamaño;
24     cout << "Ingrese el tamaño del array: ";
25     cin >> tamaño;
26
27     // Llama a la función y obtén el vector resultante
28     vector<int> vectorResultado = crearVector(tamaño);
29
30     cout << "El array retornado es: ";
31     for (size_t i = 0; i < vectorResultado.size(); ++i) {
32         cout << vectorResultado[i] << " ";
33     }
34
35     return 0;
36 }
```

Ejemplo de retorno de array usando vector en C++
Ingrese el tamaño del array: 8
El array retornado es: 0 2 4 6 8 10 12 14

Process exited after 2.874 seconds with return value 0
Presione una tecla para continuar . . .

PRESENTACIÓN

Al finalizar tu laboratorio deberás subir:

- Un archivo en formato pdf con el nombre (Deber3_Programación_NApellido)

RECOMENDACIONES

- Es preferible tener una práctica activa dentro del uso de arreglos en lo que se puede abordar problemas prácticos, participación en proyectos, fortaleciendo comprensión conceptual.
- La colaboración con compañeros de clase y la participación en sesiones de revisión de código pueden ser valiosas. Al analizar y discutir soluciones implementadas por otros, se pueden ganar perspectivas adicionales sobre enfoques y técnicas relacionadas con arreglos.

CONCLUSIONES

Utilizar arreglos en la solución de problemas, Buscando una comprensión sólida de los conceptos fundamentales relacionados con arreglos, incluyendo, pero no limitado a índices, recorridos, manipulación de elementos y dimensiones.

ENLACES

ENLACE DE GITHUB:

<https://github.com/JosueGuerra2023B/programacion2023B/tree/main/Deber%203>

ENLACE ONEDRIVE:

[Deber 3](#)

Bibliografía

- [StackOverflow, «StackOverflow,» 27 04 2020. [En línea]. Available:
1 [https://es.stackoverflow.com/questions/349765/c%C3%B3mo-pasar-arrays-a-una-funci%C3%B3n-y-](https://es.stackoverflow.com/questions/349765/c%C3%B3mo-pasar-arrays-a-una-funci%C3%B3n-y-c%C3%B3mo-retornarlos)
] [c%C3%B3mo-retornarlos](https://es.stackoverflow.com/questions/349765/c%C3%B3mo-pasar-arrays-a-una-funci%C3%B3n-y-c%C3%B3mo-retornarlos). [Último acceso: 26 01 2024].
- [J. Hu, «DelftStack,» 12 10 2023. [En línea]. Available: <https://www.delftstack.com/es/howto/cpp/how-to-return-array-from-a-function-in-cpp/#:~:text=C%C3%B3mo%20devolver%20un%20array%20desde%20una%20funci%C3%B3n%20en,d%20evolver%20el%20array%20desde%20la%20funci%C3%B3n%20en%20C%2B%2B>. [Último acceso: 26 01 2023].