22F-3298

Assignment\_1

/\*Question\_1\*/

#include<iostream>

using namespace std;

/\*Palindrome & Reverse of Matrix \*/

void reversearray(int\* array1, int\* array2, int size);

bool palindromarray(int\* arr2, int size);

int main()

{

int\* ptr1,\*ptr2;

int size;

cout << "Enter the size of an array : ";

cin >> size;

ptr1 = new int[size];

ptr2 = new int[size];

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

{

cout << "Enter Number of Array [ " << i << " ] : ";

cin >> \*(ptr1 + i);

}

reversearray(ptr1, ptr2, size);

cout << endl;

if (palindromarray(ptr2, size))

{

cout << "Array is Palindrome !";

}

else

cout << "Array is not a palindrome number!";

cout << endl;

cout <<"-----Reversed Array is------------- " << endl;

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

{

cout << \*(ptr2 + i) << " ";

}

}

void reversearray(int\* array1, int\* array2, int size)

{

int j = size - 1;

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

{

\*(array2 + i) = \*(array1 + j);

j--;

}

}

bool palindromarray(int\* arr2, int size)

{

int j = size - 1;

int count = 0;

for (int i = 0; i < size/2; i++)

{

if (\*(arr2 + i) == \*(arr2 + j))

{

count++;

}

j--;

}

if (count == size / 2)

{

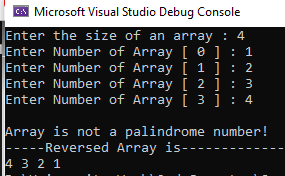
return true;

}

else

return false;

}



/\*Question\_2

Transpose of Matrix\*/

#include<iostream>

using namespace std;

int main()

{

int\*\* arr;

int\*\* transpose\_arr;

const int rows = 3, cols = 3;

arr = new int\* [rows];

transpose\_arr = new int\* [rows];

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

{

arr[i] = new int[cols];

transpose\_arr[i] = new int[cols];

}

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

{

for(int j = 0; j < cols; j++)

{

cin>> arr[i][j];

}

}

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

{

for (int j = 0; j < cols; j++)

{

transpose\_arr[j][i] = arr[i][j];

}

}

cout << endl;

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

{

for (int j = 0; j < cols; j++)

{

cout << arr[i][j] << " ";

}

cout << endl;

}

cout << endl;

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

{

for (int j = 0; j < cols; j++)

{

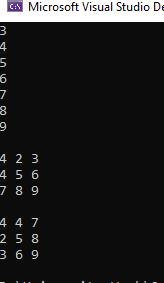
cout<< transpose\_arr[i][j]<<" ";

}

cout << endl;

}

}



/\*Qustion\_3 :

Upper Triangular Matrix\*/

#include<iostream>

using namespace std;

void upper\_half(int \*\*arr , int n , int m)

{

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

{

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

{

cout << " "<<" ";

}

for (int j = i; j < m; j++)

{

cout << arr[i][j]<<" ";

}

cout << endl;

}

}

void main()

{

int\*\* arr;

int rows,cols;

cout << "Enter the rows of an array : ";

cin >> rows;

cout << "Enter the columns of an array : ";

cin >> cols;

arr = new int\*[rows];

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

{

arr[i] = new int[cols];

}

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

{

for (int j = 0; j < cols; j++)

{

cin >> arr[i][j];

}

}

cout <<endl << "----------Original Array-----------" << endl;

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

{

for (int j = 0; j < cols; j++)

{

cout << arr[i][j] << " ";

}

cout << endl;

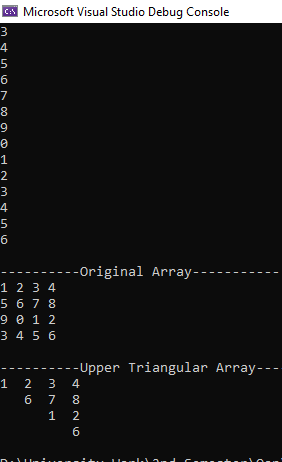
}

cout << endl;

cout << "----------Upper Triangular Array---------------" << endl;

upper\_half(arr , rows , cols);

}



/\*Question\_4\*/

#include <iostream>

using namespace std;

void shallow\_copy(int\*& destination, int\* original);

void deep\_copy(int\*& dest, int\* src, int size);

int main()

{

int size = 5;

int\* original = new int[size];

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

{

original[i] = i;

}

int\* shallow\_copy\_dest;

shallow\_copy(shallow\_copy\_dest, original);

cout << "Shallow copy:\n";

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

{

cout << shallow\_copy\_dest[i] << " ";

}

cout << endl;

int\* deep\_copy\_dest;

deep\_copy(deep\_copy\_dest, original, size);

cout << "Deep copy:\n";

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

{

cout << deep\_copy\_dest[i] << " ";

}

cout << endl;

cout << "Enter new Values in Original Array : ";

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

{

cin>>original[i];

}

cout << "After changing original:\n";

cout << "Shallow copy:\n";

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

{

cout << shallow\_copy\_dest[i] << " ";

}

cout << endl;

cout << "Deep copy:\n";

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

{

cout << deep\_copy\_dest[i] << " ";

}

cout <<endl;

return 0;

}

void shallow\_copy(int\*& destination, int\* original)

{

destination = original;

}

void deep\_copy(int\*& destination, int\* original, int size)

{

destination = new int[size];

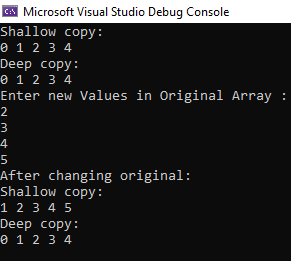
for (int i = 0; i < size; i++)

{

destination[i] = original[i];

}

}



/\*Question\_5\*/

#include<iostream>

using namespace std;

int main()

{

int N =5 , count=0,new\_size=0;

cout << "enter size of array \t : ";

cin >> N;

int\* arr = new int[N];

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

{

cout << "enter number \t";

cin >> arr[i];

count++;

if (count == N)

{

cout << "How many more elements you want to enter ? " << endl;

cout << " otherwise enter 0 if you don't want to extend the array more :\t";

cin >> new\_size;

int\* temp = new int[N + new\_size];

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

{

temp[i] = arr[i];

}

delete[] arr;

arr = temp;

N = N + new\_size;

}

}

cout << endl;

cout << "FINAL ARRAY IS : \n";

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

{

cout << arr[i]<<" ";

}

}

