Q1: Write a recursive function that computes the sum of all numbers from 1 to n, where n

is given as parameter.

Code:

#include <iostream>

using namespace std;

int Sum(int n);

void main()

{

int numb = 0;

do

{

cout << "Enter a Positive Natural Number Greater than 0: " << endl;

cin >> numb;

} while (numb <= 0);

cout << "Sum of Natural Numbers upto "<< numb <<" is: " << Sum(numb) << endl;

system("pause>0");

}

int Sum(int n)

{

if (n > 0)

{

return n + Sum(n - 1);

}

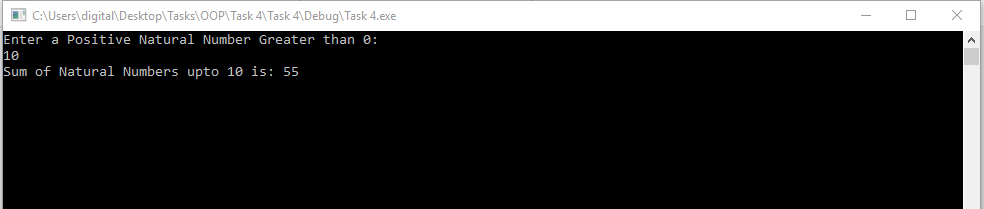
else

{

return 0;

}

}



Q2: Write a recursive definition of the function Greatest\_common\_divisor(),which takes two positive integer arguments and returns their greatest common divisor. Test your function in a suitable main program.

Code:

#include <iostream>

using namespace std;

int Greatest\_common\_divisor(int x, int y);

void main()

{

int Numb1 = 0,Numb2 = 0, HCF = 0;

do

{

cout << "Enter First Non-Zero Positive Integer: " << endl;

cin >> Numb1;

} while (Numb1 <= 0);

do

{

cout << "Enter Second Non-Zero Positive Integer: " << endl;

cin >> Numb2;

} while (Numb2 <= 0);

HCF = Greatest\_common\_divisor(Numb1, Numb2);

cout << "Greatest Common Divisor of " << Numb1 << " And " << Numb2 << " is (" << HCF << ")" << endl;

system("pause>0");

}

int Greatest\_common\_divisor(int x, int y)

{

if (y != 0)

{

return Greatest\_common\_divisor(y, x%y);

}

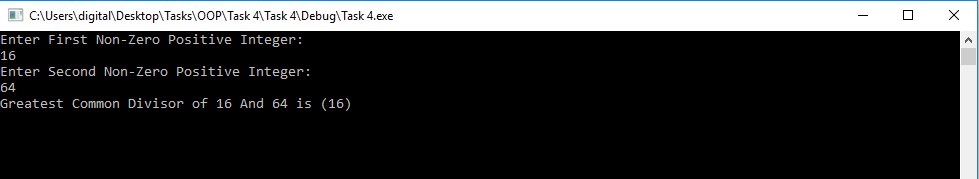
else

{

return x;

}

}



Q3: Write a recursive function that finds and returns the minimum element in an array,

where the array and its size are given as parameters.

Code:

#include <iostream>

using namespace std;

int Min(int[], int S);

void main()

{

int Size = 0;

do

{

cout << "Enter A Positive Number As Size of Array:" << endl;

cin >> Size;

} while (Size <= 0);

int \*Arr = new int[Size];

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

{

cout << "Enter Value Number " << i + 1 << ": " << endl;

cin >> Arr[i];

}

cout << endl ;

cout << "Minimum Element of Array is " << Min(Arr, Size);

system("pause>0");

}

int Min(int Arr[], int S)

{

if (S == 0)

{

return Arr[0];

}

else

{

if (Arr[S - 1] < Min(Arr, S - 1))

{

return Arr[S - 1];

}

else

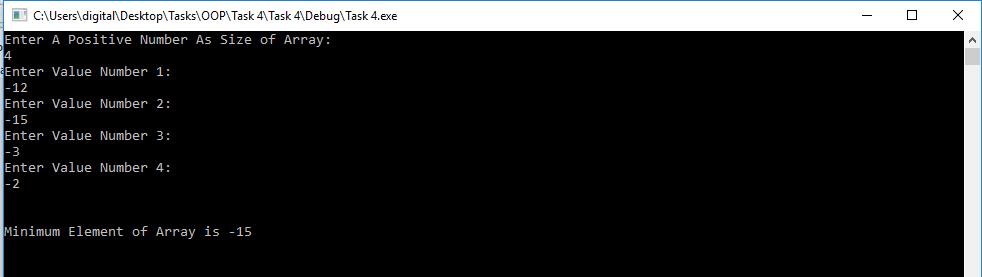
{

return Min(Arr, S - 1);

}

}

}



Q4: Write a recursive function that searches for a target in a sorted array using binary

search, where the array, its size and the target are given as parameters.

Code:

#include <iostream>

using namespace std;

int find(int [],int , int );

int main()

{

int Size = 0,Number=0;

do

{

cout << "Enter A Positive Number As Size of Array:" << endl;

cin >> Size;

} while (Size <= 0);

int \*Arr = new int[Size];

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

{

cout << "Enter Value Number " << i + 1 << ": " << endl;

cin >> Arr[i];

}

do

{

cout << "Enter The Number Which You Want To Find: " << endl;

cin >> Number;

} while (Number <= 0);

cout << endl;

if (find(Arr, Size, Number) != -1)

{

cout << "Number " << Number << " is at " << find(Arr, Size, Number) << endl;

}

else

{

cout << "Number "<<Number<<" = " << find(Arr, Size, Number) << " So number not found..." << endl;

}

system("pause>0");

}

int find(int Arr[],int S, int T)

{

if (S == 0)

{

return -1;

}

int middle = S / 2;

if (Arr[middle] == T)

{

return middle;

}

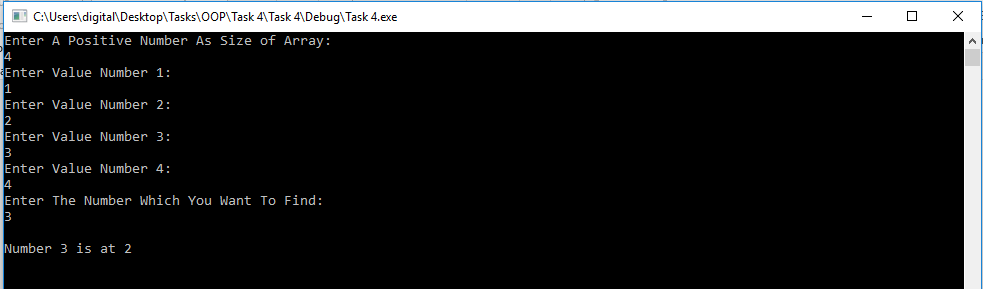
bool lower = T < Arr[middle];

int n = lower ? find(Arr, S, middle)

: find(Arr + middle + 1, S - middle - 1,T);

return n != -1 && !lower ? n + middle + 1 : n;

}



Q5: Write a recursive function to generate the following pattern of stars:

\*

\* \*

\* \* \*

\* \* \* \*

\* \* \*

\* \*

\*

Code:

#include <iostream>

using namespace std;

void Star(int num, int size);

int main()

{

int Numb = 0;

cout << "Enter A Positive Number: ";

cin >> Numb;

Star(1, Numb);

system("pause>0");

}

void Star(int num,int size)

{

if (num > size)

{

return;

}

else

{

for (int q = 0; q < num; q++)

{

cout << "\*";

}

cout << endl;

Star(num+1, size);

for (int q = 1; q < num; q++)

{

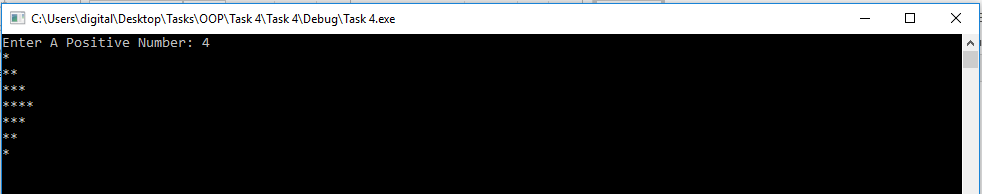
cout << "\*";

}

cout << endl;

}

}



Q6: Write a recursive function which check whether the string is palindrome or not.

Your function should be of type Boolean.

Code:

#include <iostream>

#include <string>

using namespace std;

bool Pallindrom(string);

int main()

{

string word;

cout << "Enter Word : " << endl;

cin >> word;

if (Pallindrom(word) == true)

{

cout << word << " is a Pallindrom" << endl;

}

else

{

cout << word << " is not a Pallindrom" << endl;

}

system("pause>0");

}

bool Pallindrom(string s)

{

int len = s.length();

char start = s.at(0);

char last = s.at(len-1);

if (len > 1)

{

last = s.at(len-1);

}

if (start == last && len <= 2)

{

return true;

}

else if (start != last)

{

return false;

}

else

{

s = s.substr(1, s.size() - 2);

return Pallindrom(s);

}

}

