Assignment - 24 (Functions in C++)

1. Define a function to check whether a given number is a Prime number or not.

#include<iostream>

using namespace std;

int isprime(int);

int isprime(int num)

{

if(num < 2)

return 0;

else

{

for(int i = 2; i <= num / 2; i++)

if(num % i == 0)

return 0;

return 1;

}

}

int main()

{

int number;

cout<<"Enter a number to check whether the number is prime or not: ";

cin>>number;

if(isprime(number))

cout<<number<<" is a prime number.";

else

cout<<number<<" is not a prime number.";

return 0;

}

2. Define a function to find the highest value digit in a given number.

#include<iostream>

using namespace std;

int highestValueDigit(int number)

{

int highestValue = -1, digit;

if(number == 0)

return 0;

if(number < 0)

number = -number;

while(number)

{

digit = number % 10;

if(digit > highestValue)

highestValue = digit;

number = number / 10;

}

return highestValue;

}

int main()

{

int number;

cout<<"Enter a number to find its highest value digit: ";

cin>>number;

cout<<"Highest value digit in "<<number<<" is "<<highestValueDigit(number);

return 0;

}

3. Define a function to calculate x raised to the power y.

#include<iostream>

using namespace std;

double baseRaisedToPower(double base, int power)

{

if(base < 0)

{

if(power % 2)

return -1 \* baseRaisedToPower(base \* - 1, power);

else

return baseRaisedToPower(base \* -1, power);

}

if(base == 0)

return 0;

else if(power == 0)

return 1;

else

return base \* baseRaisedToPower(base, power - 1);

}

int main()

{

int power;

double base;

cout<<"Enter the base and power to find base raised to power: ";

cin>>base>>power;

cout<<base<<" raised to the power "<<power<<" is "<<baseRaisedToPower(base, power);

return 0;

}

4. Define a function to print Pascal Triangle up to N lines.

#include<iostream>

using namespace std;

void printPascalTriangle(int N)

{

int arr[N][2 \* N - 1], i, j;

// initializing all the values of 2d array with 0

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

{

for(j = 0; j < 2 \* N - 1; j++)

{

arr[i][j] = 0;

}

}

// building pascal triangle

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

{

for(j = 0; j < 2 \* N - 1; j++)

{

if(j >= N - 1 - i && j <= N - 1 + i)

{

if(N % 2)

{

if(i % 2)

{

if(j % 2)

arr[i][j] = arr[i - 1][j - 1] + arr[i - 1][j + 1];

}

else if(i == 0)

{

arr[i][j] = 1;

}

else if(i == N - 1 && (j == 0 || j == 2 \* N - 2))

arr[i][j] = 1;

else

{

if(j % 2 == 0)

arr[i][j] = arr[i - 1][j - 1] + arr[i - 1][j + 1];

}

}

else

{

if(i % 2)

{

if(i == N - 1 && (j == 0 || j == 2 \* N - 2))

arr[i][j] = 1;

else if(j % 2 == 0)

arr[i][j] = arr[i - 1][j - 1] + arr[i - 1][j + 1];

}

else if(i == 0)

{

arr[i][j] = 1;

}

else

{

if(j % 2)

arr[i][j] = arr[i - 1][j - 1] + arr[i - 1][j + 1];

}

}

}

}

}

// printing pascal triangle

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

{

for(j = 0; j < 2 \* N - 1; j++)

{

if(arr[i][j])

cout<<arr[i][j];

cout<<"\t";

}

cout<<"\n\n";

}

}

int main()

{

printPascalTriangle(6);

return 0;

}

5. Define a function to check whether a given number is a term in a Fibonacci series or not.

#include<iostream>

using namespace std;

int isNumberPresentInFibonacciSeries(int number)

{

int f1 = 0, f2 = 1, f3 = -1;

if(number < 0)

return 0;

else if(number == 0)

return 1;

else

{

while(f3 < number)

{

f3 = f1 + f2;

f1 = f2;

f2 = f3;

}

if(f3 == number)

return 1;

else

return 0;

}

}

int main()

{

int number;

cout<<"Enter a number to check whether it is a term in fibonacci series or not: ";

cin>>number;

if(isNumberPresentInFibonacciSeries(number))

cout<<number<<" is present in fibonacci series.";

else

cout<<number<<" is not present in fibonacci series.";

return 0;

}

6. Define a function to swap data of two int variables using call by reference

#include<iostream>

using namespace std;

void swapNums(int &num1, int &num2)

{

num1 = num1 + num2;

num2 = num1 - num2;

num1 = num1 - num2;

}

int main()

{

int a = 3, b = 7;

cout<<"Numbers before swapping are:\na = "<<a<<" and b = "<<b<<endl;

swapNums(a, b);

cout<<"Numbers after swapping are:\na = "<<a<<" and b = "<<b<<endl;

return 0;

}

7. Write a function using the default argument that is able to add 2 or 3 numbers.

#include<iostream>

using namespace std;

int sum(int, int, int = 0);

int main()

{

cout<<"1 + 2 = "<<sum(1, 2)<<endl;

cout<<"1 + 2 + 3 = "<<sum(1, 2, 3)<<endl;

return 0;

}

int sum(int num1, int num2, int num3)

{

return num1 + num2 + num3;

}

8. Define overloaded functions to calculate area of circle, area of rectangle and area of triangle

#include<iostream>

using namespace std;

double area(int);

int area(int, int);

double area(double, double);

// area of circle

double area(int radius)

{

return 3.14 \* radius \* radius;

}

// area of rectangle

int area(int length, int breadth)

{

return length \* breadth;

}

// area of triangle

double area(double base, double height)

{

return 0.5 \* base \* height;

}

int main()

{

int radius = 9, length = 3, breadth = 4;

double base = 4, height = 9;

cout<<"Area of circle is "<<area(radius)<<endl;

cout<<"Area of rectangle is "<<area(length, breadth)<<endl;

cout<<"Area of triangle is "<<area(base, height)<<endl;

return 0;

}

9. Write functions using function overloading to find a maximum of two numbers and both the numbers can be integer or real.

#include<iostream>

using namespace std;

int greaterNumber(int num1, int num2)

{

if(num1 > num2)

return num1;

else

return num2;

}

float greaterNumber(float num1, float num2)

{

if(num1 > num2)

return num1;

else

return num2;

}

float greaterNumber(float num1, int num2)

{

if(num1 > num2)

return num1;

else

return num2;

}

float greaterNumber(int num1, float num2)

{

if(num1 > num2)

return num1;

else

return num2;

}

int main()

{

int num1 = 3, num2 = 4;

float num3 = 3.7, num4 = 4.3;

cout<<"Greater number between int "<<num1<<" and int "<<num2<<" is: "<<greaterNumber(num1, num2)<<endl;

cout<<"Greater number between float "<<num3<<" and float "<<num4<<" is: "<<greaterNumber(num3, num4)<<endl;

cout<<"Greater number between float "<<num3<<" and int "<<num1<<" is: "<<greaterNumber(num3, num1)<<endl;

cout<<"Greater number between int "<<num2<<" and float "<<num4<<" is: "<<greaterNumber(num2, num4)<<endl;

return 0;

}

10. Write functions using function overloading to add two numbers having different data types.

#include<iostream>

using namespace std;

int add(int num1, int num2)

{

return num1 + num2;

}

float add(float num1, float num2)

{

return num1 + num2;

}

float add(float num1, int num2)

{

return num1 + num2;

}

float add(int num1, float num2)

{

return num1 + num2;

}

int main()

{

int num1 = 3, num2 = 4;

float num3 = 3.7, num4 = 4.3;

cout<<"Sum of "<<num1<<" and "<<num2<<" is: "<<add(num1, num2)<<endl;

cout<<"Sum of "<<num3<<" and "<<num4<<" is: "<<add(num3, num4)<<endl;

cout<<"Sum of "<<num3<<" and "<<num1<<" is: "<<add(num3, num1)<<endl;

cout<<"Sum of "<<num2<<" and "<<num4<<" is: "<<add(num2, num4)<<endl;

return 0;

}