Question 1:

An approximate value of pi can be calculated using the series given below:

$pi = 4 [1 – 1/3 + 1/5 – 1/7 + 1/9 ... + ((–1)^n)/(2n + 1)]$

Write a C++ program that calculates and outputs the value of pi using the series above. The program takes an input n that determines the number of terms in the approximation of the value of pi and outputs the approximation.

Your program should use a while loop that allows the user to repeat this calculation for new values n until the user enters a negative value for n to end the program.

Answer:

```
#include<iostream>
#include<math.h>
#include<iomanip>
using namespace std;

int main(){

double n, a, b, c, pi;

cout << "Please enter n value > " << endl;
cin >> n;

do{
pi = 1;
c = 1;

for(c>=0; c<=n; c++){
   a = pow(-1, c);
   b = (2 * c + 1);
   pi = pi + (a / b);
}
   pi = pi * 4;
cout << "The pi value is: " << setprecision(10) << pi << endl << endl;
cout << "Please enter n value > " << endl;
cin >> n;
}while(n > 0);

if (n < 0){
   cout << "Invalid input, program ends." << endl;

return 0;
}
```

Question 2:

Write a program to get two integers n >0 and k > 1. Whenever at least one of the two inputs is not valid, request both inputs again. Only when both inputs are valid, calculate the numerical of value of the following sum:

nsin(πn)+π33!(1n)2−π55!(1n)4+...+(−1)k+1π2k+1(2k+1)!(1n)2knsin(πn)+π33!(1n)2−π55!(1n)4+...+(−1)k+1π2k+1(2k+1)!(1n)2k

To implement a loop to calculate the sum, you need to identify the relationship between

- the terms π33!(1n)2π33!(1n)2 and −π55!(1n)4−π55!(1n)4
- and the terms −π55!(1n)4−π55!(1n)4 and π77!(1n)6π77!(1n)6
- ...

Use a **do ... while** loop to validate the inputs and a **for** loop to calculate the sum. Your code must not use any selection.

Answer:

```
#include <stdio.h>
#include<iostream>
#include<math.h>
#include<iomanip>
using namespace std;
int fact(int f) {
  if ((f==0)||(f==1))
    return 1;
  else
    return f*fact(f-1);
}
int main(){
for(;;){
   int n, k, a, y, z;
   double b, c, d, e, f, g, h;
   /*
   a = counter
   b = odd #
   c = even #
   d = the formula
   e = result
```

```
    f = factorial
    g = 1 / n
    h = n sin(pi/n)
    y = a + 1
    z = 1
    */
    const double pi = 3.1415926;//53589793238462643383279;
    z = 1;
    do{
        cout << "Please enter n & k values, note that n>0 and k>1: " << endl;
        cin >> n >> k;
        cout << endl;
    } while(n <= 0 || k <= 1);

    do{
        g = 1 / n;
        h = n * sin(pi / n);
        e = h;
        a = 1;
        z++;

        for (a >= 1; a <= k; a++){
            b = 2 * a + 1;
            c = 2 * a;
            f = b;
            y = a + 1;
            d = pow(-1, y) * pow(pi, b) / fact(f) * pow(g, c);
            e = e + d;
        }
        cout << "The answer is: " << e << endl;
    } while(n>0 && k>1 && z == 1);
}
return 0;}
```

Question 3:

Given an integer n (with $1 \leq n \leq 40$), we want to produce a program that displays the equivalent of n, in the roman numerals. Assume that the user will always enter a valid value for n.

To get full mark, you solution must use at most 2 loops (while or for loops) and at most two selections (switch case or if ... else if ... else if)

1	2	3	4	5	6	7	8	9	10
I	II	III	IV	V	VI	VII	VIII	IX	X
11	12	13	14	15	16	17	18	19	20
XI	XII	XIII	XIV	XV	XVI	XVII	XVIII	XIX	XX
21	22	23	24	25	26	27	28	29	30
XXI	XXII	XXIII	XXIV	XXV	XXVI	XXVII	XXVIII	XXIX	XXX
31	32	33	34	35	36	37	38	39	40
XXXI	XXXII	XXXIII	XXXIV	XXXV	XXXVI	XXXVII	XXXVIII	XXXIX	XL

Answer:

```
#include <stdio.h>
#include<iostream>
#include<iomanip>
#include<math.h>
using namespace std;


int main(){
   for (;;){
      int n, m;
      string roman[] = {"I", "II", "III", "IV", "V", "VI", "VII", "VIII", "IX", "X","XI", "XII",
"XIII", "XIV", "XV", "XVI", "XVII", "XVIII", "XIX", "XX","XXI", "XXII", "XXIII", "XXIV",
"XXV", "XXVI", "XXVII", "XXVIII", "XXIX", "XXX","XXXI", "XXXII", "XXXIII",
"XXXIV", "XXXV", "XXXVI", "XXXVII", "XXXVIII", "XXXIX", "XL"};
   cout << "Please enter a positive number smaller than 40 to convert into R.numerals: " << endl;
   cin >> n;
   m = n - 1;
   cout << roman[m] << endl;
   }
   return 0;
}
```