|  |  |
| --- | --- |
| **Data Structures & Algorithms**  Diploma in IT, ISF, FI  Year 2 (2018/19) Semester 4 | **Week 6** |
| **2 Hours** |
| **Tutorial 6 – Recursion** | |

1. Write a recursive function that will compute the sum of the first n integers in an array of at least n integers.

Hint: begin with the nth integer.

int sumArray(int arr[], int index)

{

if (index > 0)

return arr[index-1] + sumArray(arr, index-1);

else

return 0;

}

OR

int sumArray(int arr[], int index)

{

if (index == 0)

return 0;

else

return arr[index-1] + sumArray(arr, index-1);

}

2. Describe the problem with the following recursive function:

void printNum(int n)

{

cout << n <<endl;

printNum(n-1);

}

The recursive function will keep running because there is no base case to end the recursion.

3. Given an integer n > 0, write a recursive function that returns the sum of 1 through n.

int sum(int n)

{

if(n == 1)

return 1;

else

return n + sum(n-1);

}

4a. Write a recursive C++ function writeLine that writes a character repeatedly to form a line of n characters. For example, writeLine(‘\*’, 5) produces the line \*\*\*\*\*

void writeLine(char c, int numOfTimes)

{

if (numOfTimes == 1)

cout << c << endl;

else

{

cout << c;

writeLine(c, numOfTimes - 1);

}

}

b. Now write a recursive function writeBlock that uses writeLine to write m lines of n characters each. For example, writeBlock(‘\*’, 3, 5) produces the output

\*\*\*\*\*

\*\*\*\*\*

\*\*\*\*\*

void writeBlock(char c, int numOfRows, int numOfTimes)

{

if (numOfRows == 1)

writeLine(c, numOfTimes);

else

{

writeLine(c, numOfTimes);

writeBlock(c, numOfRows - 1, numOfTimes);

}

}

5. Consider the following program:

int f(int n);

int main()

{

cout<< “The value of f(8) is “ << f(8) << endl;

return 0;

}

// pre: n >= 0

int f(int n)

{

cout << “Function entered with n = “ << n << endl;

switch(n)

{

case 0 :

case 1 :

case 2 : return n + 1;

default: return f(n-2) \* f(n-4);

}

}

Show the exact output of the program. What argument values, if any, could you pass to the function f to cause an infinite recursion?

**Output:**

Function entered with n = 8

Function entered with n = 6

Function entered with n = 4

Function entered with n = 2

Function entered with n = 0

Function entered with n = 2

Function entered with n = 4

Function entered with n = 2

Function entered with n = 0

The value of f(8) is 27

**Values that can cause an infinite recursion:**

Odd numbers that are more than 1

Eg: 3, 5, 7, 9…