Faculty of Science and Engineering

COMP125 Fundamentals of Computer Science
Workshop Week 9

Learning outcomes

By the end of this session, you will have learnt about recursions.

1. **Recursion trace**

   Consider the following recursive function definition,

```
int foo(int a) {
        if(a == 2)
                return 2;
        return a + foo(a / 2);
}
```

   What is the value of variable `result` if the function call is,

```
        int result = foo(16);
```

2. **Debugging recursive functions**

   The following function attempts to compute the factorial of integer n. What is wrong with the function?

```
int factorial(int n) {
        return n * factorial(n - 1);
        if(n == 0)
                return 0;
}
```

3. **Debugging recursive functions**

   Give an example of a value, that, if passed to the function `foo` from the previous question, calls itself indefinitely.

4. **Some more recursive trace**

   Consider the following recursive function definition,

```
int foo(int a) {
        if(a <= 0)
                return 0;
        if(a % 2 == 0)
                return foo(a/2);
        else
                return 1 + foo(a/2);
}
```

What is the value of variable `result` if the function call is,

```
        int result = foo(59);
```

5. **Writing a recursive function**

   Write a recursive function, that when passed an integer, returns the number of even digits in that integer. Return 0 if the integer is 0.

6. **Writing a recursive function**

   Write a recursive function, that when passed an integer *n*, return the sum of squares of the first *n* positive integers $(1+2+...+n)$.

7. **Writing a recursive function dealing with text**

   Write a recursive function, that when passed a String, returns the number of digits in the String.

8. **Counting recursive function calls**

   How many calls are made to `gcd` if the original call is `gcd(30, 72`?

```
int gcd(int a, int b) {
        if(a < b)
                return gcd(b, a);
        if(b == 0)
                return a;
        return gcd(b, a%b);
```

9. **(Tracing slightly more complex recursive functions)**

   Consider the definition of the following recursive function,

```
public static void displayBrackets(int n) {
        if(n == 0)
                return;
        System.out.print("{");
        for(int i=0; i < n - 2; i++) {
                displayBrackets(n - 1);
                System.out.print(", ");
        }
        displayBrackets(n - 1);
        System.out.print("}");
}
```

   What is the output of the following statement?

```
displayBrackets(3);
```

10. **(Assessed task) Defining recursive functions**

    I have made up a sequence called a *tribonacci* sequence. The first three numbers of this sequence are 1, 2 and 3, and every subsequent number in this sequence is the sum of the previous **three** numbers. Thus, the sequence is $1,2,3,6,11,20,37,68,....$ Write a function to compute the n^{th} *tribonacci* number. Assuming the 1^{st} number is 1.

11. **(Assessed task) Counting recursive function calls**

    How many calls are made to `tribonacci` if the original call is `tribonacci(5)`?

12. **(Voluntary Assessed task) Writing a recursive function**

    Write a recursive function that displays an hour-glass pattern. For example, it displays the following pattern for $n = 5$.

```
***********
 *********
  *******
   *****
    ***
     *
     *
    ***
   *****
  *******
 *********
***********
```

And it displays the following pattern for $n = 7$.

```
***************
 *************
  ***********
   *********
    *******
     *****
      ***
       *
       *
      ***
     *****
    *******
   *********
  ***********
 *************
***************
```