**Concept part**

|  |
| --- |
| **Name:** Elaf Yousef Aloufi |
| **ID:** 1911265 |
| **Section:** BBR |

**Question1.** Consider the following recursive method:

**import** java.util.**\***;

**public class** code {

**public** **static** **int** recFunction (**int** A[])

{

**if**(A.length == 1)

**return** A[0];

**int** m = A.length %2;

**int** val1 = recFunction(**Arrays**.copyOfRange(A, 0, m + 1));

**int** val2= recFunction(**Arrays**.copyOfRange(A, m+1, A.length));

**if** (val1 > val2)

**return** val1;

**return** val2;

}

**public** **static** **void** main(**String** args[])

{

**int** A[] = {21, 6, 145, 26, -2, 0, 19, 243, 15};

**System**.out.println(recFunction(A));

}

}

1. Trace the above code (15 pts)

recFunction({21,6,145, 26, -2, 0, 19, 243, 15});

return 243;

recFunction({21,6});

return 21;

recFunction({145, 26, -2, 0, 19, 243, 15});

return 243;

recFunction({145, 26});

return 145;

recFunction({-2, 0, 19, 243, 15});

return 243;

recFunction({21});

return 21;

recFunction({6});

return 6;

recFunction({145});

return 145;

recFunction({26});

return 26;

recFunction({-2, 0});

return 0;

recFunction({19, 243, 15});

return 243;

recFunction({0});

return 0;

recFunction({-2});

return -2;

recFunction({19, 243});

return 243;

recFunction({15});

return 15;

recFunction({19});

return 19;

recFunction({243});

return 243;

Result= 243

1. What does the method recFunction calculate? (5 pts)

The largest number in the array.

**Note**: You must write your tracing using recursion trace - its format is described in the course lectures. You must also show the final answer that is returned.

**Question 2.** Consider a recursive method that removes all occurrences of substring in a given string

**Examples:**

Input1 : str=‘abcccabc’, substr= ‘abc’, result1: ‘cc’

Input2 : str=‘aabcbcc’, substr= ‘abc’, result2: ‘abcc’

Input3 : str=‘bacbcbcc’, substr= ‘ab’, result3: ‘bacbcbcc’

1. Write a complete pseudo code/algorithm to solve the requirement of the method as explained above. (15 points)

**Note:** write the information that you need to describe the header of the method

**Input:**

String str, String substr

**Output:**

String

**Method header:**

public static String remove(String str, String substr)

**Algorithm:**

**Step1:**

if str.length() >= substr.length()

**Step2:**

if str.substring(0, substr.length()).equals(substr)

return remove(str.substring(substr.length()), substr);

END IF

**Step3:**

else

return str.substring(0, 1) + remove(str.substring(1), substr);

END ELSE

END IF

**Step4:**

return str;

1. Trace your algorithm using input1. (5 points)

**Note:** You must write your tracing using recursion trace - its format is described in the course lectures. You must also show the final answer.

remove(“abcccabc”,”abc”);

remove(“ccabc”,”abc”);

return “cc”;

remove(“cabc”,”abc”);

return “c”;

remove (“abc”,”abc”);

return “”;

remove(“”,”abc”);

return “”;

**Input 1:**

**str= “abcccabc”**

**substr= “abc”**

Result= “cc”