1

# Assignment 6 - Recursion

1. Take as input N, a number. Print odd numbers in decreasing sequence (up until

0) and even numbers in increasing sequence (up until N). E.g. for N = 6 print 5, 3, 1, 2, 4.

1. Take as input N, a number. Print the following pattern (for N = 5)

## \*

**\* \***

**\* \* \***

**\* \* \* \***

**\* \* \* \* \***

1. Take as input N, a number. Print the following pattern (for N = 5)

## \* \* \* \* \*

**\* \* \* \***

**\* \* \***

**\* \***

**\***

1. Take as input N, a number. Print the following pattern (for N = 6)

## 1

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **1** | **1** |  |  |  | |
| **1** | **2** | **1** |  |
| **1** | **3** | **3** | **1** |
| **1** | **4** | **6** | **4** | **1** |  |
| **1** | **5** | **10** | **10** | **5** | **1** |

1. Take as input N, a number. Write a recursive function to find Nth triangle where 1st triangle is 1, 2nd triangle is 1 + 2 = 3, 3rd triangle is 1 + 2 + 3 = 6, so on and so forth. Print the value returned.
2. Take as input N, the size of array. Take N more inputs and store that in an array. Write a recursive function which returns true if the array is sorted and false otherwise. Print the value returned.
3. Take as input N, the size of array. Take N more inputs and store that in an array. Take as input M, a number. Write a recursive function which returns true if M is contained in the array and false otherwise. Print the value returned.
4. Take as input N, the size of array. Take N more inputs and store that in an array. Take as input M, a number. Write a recursive function which returns the first index at which M is found in the array and -1 if M is not found anywhere. Print the value returned.
5. Take as input N, the size of array. Take N more inputs and store that in an array. Take as input M, a number. Write a recursive function which returns the last index at which M is found in the array and -1 if M is not found anywhere. Print the value returned.
6. Take as input N, the size of array. Take N more inputs and store that in an array.

2

Take as input M, a number. Write a recursive function which returns an array containing indices of all items in the array which have value equal to M. Print all the values in the returned array.

1. Take as input N, a number. Take N more inputs and store that in an array. Write a recursive function which tests if the array is a palindrome and returns a Boolean value. Print the value returned.
2. Take as input N, a number. Take N more inputs and store that in an array. Write a recursive function which reverses the array. Print the values of reversed array.
3. Take as input N, a number. Take N more inputs and store that in an array. Write a recursive function which inverses the array. Print the values of inverted array.
4. Take as input N, the size of array. Take N more inputs and store that in an array. Write a recursive function which bubble sorts the given array. Print all the values in the sorted array.
5. Take as input N, the size of array. Take N more inputs and store that in an array. Write a recursive function which selection sorts the given array. Print all the values in the sorted array.
6. Take as input str, a number in form of a string. Write a recursive function to find the sum of digits in the string. Print the value returned.
7. Take as input str, a number in form of a string. Write a recursive function to convert the number in string form to number in integer form. E.g. for “1234” return 1234. Print the value returned.
8. Take as input str1 and str2, both strings. Write a function which tests if str2 is reverse of str1 or not and returns a Boolean value. Print the value returned.
9. Take as input str, a string. Write a function which tests if the string is a palindrome or not and returns a Boolean value. Print the value returned.
10. Take as input str, a string. Write a recursive function which returns a new string in which all duplicate consecutive characters are separated by a ‘\*’. E.g. for “hello” return “hel\*lo”. Print the value returned.
11. Take as input str, a string. Write a recursive function which returns a new string in which all duplicate consecutive characters are reduced to a single char. E.g. for “hello” return “helo”. Print the value returned.
12. Take as input str, a string. Write a recursive function which moves all ‘x’ from the string to its end. E.g. for “abexexdexed” return “abeedeedxxx”. Print the value returned.
13. Take as input str, a string.
    * Write a recursive function which counts the number of times ‘hi’ appears

in the string. Print the value returned.

* + Write a recursive function which removes all ‘hi’ in the string. Print the

value returned.

* + Write a recursive function which replaces all ‘hi’ in the string with ‘bye’.

Print the value returned.

1. Take as input str, a string.
   * Write a recursive function which counts the number of times ‘hi’ appears in the string – but skip all such ‘hi’ which are followed by ‘t’ to form ‘hit’. Print the value returned.
   * Write a recursive function which removes all ‘hi’ in the string – but skip all

such ‘hi’ which are followed by ‘t’ to form ‘hit’. Print the value returned.

* + Write a recursive function which replaces all ‘hi’ in the string with ‘bye’ – but skip all such ‘hi’ which are followed by ‘t’ to form ‘hit’. Print the value returned.

1. Take as input str, a string. A “twin” is defined as two instances of a char separated by a char. E.g. "AxA" the A's make a “twin”. “twins” can overlap, so "AxAxA" contains 3 “twins” - 2 for A and 1 for x. Write a function which recursively counts number of “twins” in a string. Print the value returned.
2. Take as input str, a string
3. Write a recursive function which counts, the number of times “aaa” appears in the string. Print the value returned.

b. Write a recursive function which counts, the number of times “aaa” appears in the string, but only such instances of “aaa” should be considered which do not overlap. Print the value returned.

1. Take as input str, a string. Assume that value of a=1, b=2, c=3, d=4, z=26. Write

a recursive function (return type void) to print all possible codes for the string. E.g.

for “1123” possible codes are aabc, kbc, alc, aaw, kw.

1. Take as input str, a string. Write a recursive function that checks if the string was generated using the following rules and returns a Boolean value:
   1. the string begins with an 'a'
   2. each 'a' is followed by nothing or an 'a' or "bb"
   3. each "bb" is followed by nothing or an 'a' Print the value returned.
2. Take as input str, a string. The string contains a single pair of parenthesis “(..)”.

Write a recursive function which returns the contents between the parenthesis.

E.g. for “xyz(abc)def” return “abc”. Print the value returned.

1. Take as input str, a string. The string is a mathematical expression e.g. “[a + {b + (c + d) + e} + f]”. Write a recursive function which tests if the brackets in expression are balanced or not and returns a Boolean value. Print the value returned.