2018 SS

Question 25

[1 mark] Which of the following is not an important property of a recursive function?

(a) A recursive function must be more efficient than using a loop
(b) A recursive function calls itself in its definition
(c) A recursive function contains a base case that enables recursive calls to stop
(d) Each recursive function call solves an identical but smaller problem
(e) None of the above

Question 26

[1 mark] Consider the following recursive function definition:

```
def hailstone(value):
    print(value, end = ' ')
    if(value == 1):
        return
    elif(value % 2 == 0):
        hailstone(value // 2)
    elif(value % 2 == 1):
        hailstone(value*3 + 1)
```

What would be the output of the following code?

```
hailstone(5)
```

(a) `5 24 12 6 3 2 1`
(b) `5 8 4 2 1`
(c) `16 8 4 2 1`
(d) `5 16 8 4 2 1`
(e) None of the above

Question 27

[1 mark] Consider the following recursive function definition:

```
def fun(a,b):
    if a == b:
        return a
    else:
        return a + fun(a+1, b)
```

What would be the output of the following function call?

```
print(fun(3,9))
```

(a) `30`
(b) `42`
(c) `33`
(d) `39`
(e) None of the above

Question 28

[1 mark] What is the Big-O time complexity of the following function (`fib`) that calculates the n^{th} number in the Fibonacci sequence?

```
def fib(n):
    if n <= 2:
        return 1
    else: return fib(n-1) + fib(n-2)
```

(a) $O(2^n)$
(b) $O(n^2)$
(c) $O(n)$
(d) $O(\log n)$
(e) None of the above

Question 29
[1.5 marks] Consider the following recursive function definition:

```
def print_recursive(s):
    if len(s) == 0:
        print('Complete', end = ' ')
    else:
        print(s[-1], end = ' ')
        print_recursive(s[0:-1])
```

What would be the output of the following function call?

```
print_recursive('cs105')
```

(a) `Complete 5 0 1 s c`
(b) `Complete c s 1 0 5`
(c) `c s 1 0 5 Complete`
(d) `5 0 1 s c Complete`
(e) None of the above

Question 30
[1.5 marks] The following recursive function, `sum_list()`, takes in a list of integers and returns the sum of these values. The else block definition is missing and has been replaced with '???'

```
def sum_list(values):
    if len(values) == 0:
        return 0
    else:
        ???
```

Which statement should replace the '**???**' to correctly complete the function definition?

(a) `return values[-1] + sum_list(values[1:])`
(b) `return values[0] + sum_list(values[:-1])`
(c) `return values[-1] + sum_list(values[:-1])`
(d) `return values[1] + sum_list(values[1:])`
(e) None of the above

2017 S2
Question 19
[2 marks] Consider the following recursive function definition.

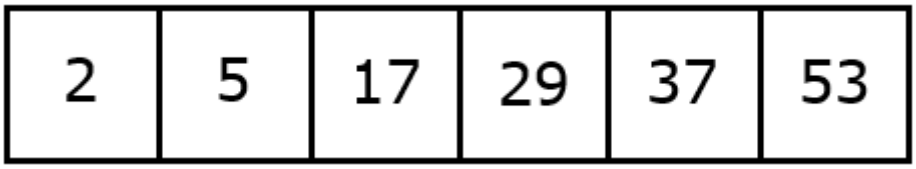
```
def exam_function(number):
    if number > 0:
        remainder = number % 2
        digit = str(remainder)
        return exam_function(number // 2) + digit
    else:
        return ""
```

What would be the output of the following function call?

```
print(exam_function(37))
```

(a) Infinite recursion.
(b) 101001
(c) 100101
(d) 010110
(e) None of the above.

Question 20
[2 marks] Consider the following list below:



If we were to use the `binary_search()` function discussed in lectures to search for the value **53** in this list, how many calls to the `binary_search()` function would be made in total (including the top level call)?

(a) 3
(b) 4
(c) 5
(d) 2
(e) None of the above.

Question 21
[2 marks] Which of the following statements is **TRUE**?

I A recursive solution to a problem is always preferable.
II A recursive function calls itself.
III Each recursive call diminishes the size of the problem.
IV A recursive function can have one or more base cases.
V A recursive function can have one or more recursive steps.

(a) II, III and IV
(b) I, II and V
(c) II, IV and V
(d) I, II, III, IV and V
(e) II, III, IV and V

Question 22
[2 marks] The `remove_vowel()` function takes a string as a parameter and returns the string with all vowels removed. For example:

```
remove_vowel("television")
```

would return the string `"tlvsn"`.

The code for the `remove_vowel()` function is provided below. The "if" block of the function definition is missing, and has been replaced with `????`.

```
def remove_vowel(a_string):
    vowels = ["a","e","i","o","u"]
    for i in range(len(a_string)):
        if a_string[i] in vowels:
            ????
    return a_string
```

Which statement should replace the `????` above to correctly complete this **recursive function** definition?

(a) `return remove_vowel(a_string[:i]) + a_string[i+1:]`
(b) `return a_string[i] + remove_vowel(a_string[:i] + a_string[i+1:])`
(c) `return a_string[i] + remove_vowel(a_string[i+1:])`
(d) `return a_string[:i] + remove_vowel(a_string[i+1:])`
(e) None of the above.

Question 37
a) In mathematics, the Pell numbers are an infinite sequence of integer values. The first 6 Pell numbers are 0, 1, 2, 5, 12, and 29. The Pell numbers can be defined recursively as follows:

$$\text{pell}(n) = \begin{cases} 0 & \text{if } n = 0 \\ 1 & \text{if } n = 1 \\ 2 * \text{pell}(n-1) + \text{pell}(n-2) & \text{if } n > 1 \end{cases}$$

In other words, the sequence of Pell numbers starts with 0 and 1, and then each Pell number is the sum of twice the previous Pell number and the Pell number before that.

Complete the `pell()` function below that takes a single integer parameter `n`, and returns the n^{th} Pell number in the sequence. The `pell()` function must be implemented recursively using the provided definition.

```
def pell(n):
```

(5 marks)

2017 S1

The following 3 questions use the `radix_convert_to_Dec(num, radix)` function below:

```
def radix_convert_to_Dec(num, radix):
    a = num // 10
    b = num % 10
    if (a > 0):
        result = b + radix * radix_convert_to_Dec(a, radix)
    else:
        result = b
    return result
```

Question 25

[1.5 marks] Which output is produced when the statement **`print(radix_convert_to_Dec(111, 2))`** is executed?

(a) 11
(b) 7
(c) 3
(d) 10
(e) 9

Question 26

[1.5 marks] Which output is produced when the statement **`print(radix_convert_to_Dec(31, 4))`** is executed?

(a) 12
(b) 11
(c) 14
(d) 10
(e) 13

Question 27

[1.5 marks] Which output is produced when the statement **`print(radix_convert_to_Dec(141, 6))`** is executed?

(a) 60
(b) 71
(c) 61
(d) 56
(e) 62

The following 3 questions use the `Dec_convert_to_radix(num, radix)` function below:

```
def Dec_convert_to_radix(num, radix):
    a = num // radix
    b = num % radix
    if (a > 0):
        result = b + 10 * Dec_convert_to_radix(a, radix)
    else:
        result = b
    return result
```

Question 32
[1.5 marks] Which output is produced when the statement **`print(Dec_convert_to_radix(24, 3))`** is executed?

(a) 222
(b) 221
(c) 210
(d) 220
(e) 211

Question 33
[1.5 marks] Which output is produced when the statement **`print(Dec_convert_to_radix(37, 4))`** is executed?

(a) 222
(b) 212
(c) 220
(d) 221
(e) 211

Question 34
[1.5 marks] Which output is produced when the statement **`print(Dec_convert_to_radix(141, 11))`** is executed?

(a) 119
(b) 121
(c) 116
(d) 118
(e) 117

2017 SS
Question 29
The `gcd()` function shown below calculates the greatest common divisor of the two input numbers. Notice that a `print()` statement has been placed at the *very start* of the function definition - this will display the inputs for every function call that occurs.

```
def g(m, n):
    print(m, n, end = ' ')
    if m == n:
        return m
    elif (m > n):
        return g(m-n, n)
    else:
        return g(m, n-m)
```

If the following call is made:

```
print('Result =', end = ' ')
g(18, 12)
```

what output would be produced?

(a) `Result = 18 12 6 6`
(b) `Result = 18 12 8 2 6 2 4 2 2`
(c) `Result = 18 12 6 4 2 4 2 2`
(d) `Result = 18 12 12 4 8 4 4 4`
(e) `Result = 18 12 6 12 6 6`

Question 30
What is the efficiency, in terms of *n*, of the following function (called `fib`) that calculates the n^{th} number in the Fibonacci sequence?

```
def fib(n):
    if n <=2:
        return 1
    else:
        return fib(n-1) + fib(n-2)
```

(a) O(log n)
(b) O(n)
(c) O(n log n)
(d) $O(2^n)$
(e) $O(n^2)$

2016S2

Question 26

[2 marks] Consider the following recursive function definition.

```
def recursive_function1(s):
    if len(s) == 1:
        return int(s) * 1
    else:
        return int(s[0]) * 2 ** (len(s) - 1) + recursive_function1(s[1:])
```

What would be the output of the following function call?

```
print(recursive_function1("10110"))
```

(a) 6
(b) Infinite recursion
(c) 14
(d) 22
(e) None of the above.

Question 27

[2 marks] What is the efficiency, in terms of *n*, of the following function (`hanoi`) that solves a Towers of Hanoi puzzle with *n* discs?

```
def hanoi(n, source, destination, spare):
    if n <= 1:
        print("base case: move disk from", source, "to", destination)
    else:
        hanoi(n - 1, source, spare, destination)
        print("move disk from", source, "to", destination)
        hanoi(n - 1, spare, destination, source)
```

(a) O(n)
(b) $O(n^2)$
(c) $O(2^n)$
(d) O(log n)
(e) O(n log n)

Question 28

[2 marks] Consider the following list below:

1	7	9	23	62	75

If we were to use the `binary_search()` function discussed in lectures to search for the value **5** in this list, how many calls to the `binary_search()` function would be made in total (including the top level call)?

(a) 0
(b) 3
(c) 5
(d) 4
(e) None of the above

Question 29
[2 marks] A palindrome is a sequence of characters that reads the same backward and forward. The following **recursive function**, `make_palindrome()`, takes a string as a parameter and returns a palindrome by combining the string with a reversed copy of itself. You can assume that the string parameter will have a length of at least 1. For example:

```
make_palindrome("2016")
```

would return the string "20166102".

The code for the `make_palindrome()` function is provided below. The "else" block of the function definition is missing, and has been replaced with ????.

```
def make_palindrome(s):
    if len(s) == 1:
        return s + s
    else:
        ????
```

Which statement should replace the ???? above to correctly complete this **recursive function** definition?

(a) `return s + s[len(s) - 1::-1]`
(b) `return make_palindrome(s[1:]) + s[0]`
(c) `return s[0] + make_palindrome(s[1:len(s) - 1]) + s[len(s) - 1]`
(d) `return make_palindrome(s[:len(s) – 1]) + s[len(s) – 1]`
(e) `return s[0] + make_palindrome(s[1:]) + s[0]`

Question 30
[2 marks] Given the recursive function below, what is the value returned by

```
recursive_function2(3,4)?
```

```
def recursive_function2(num1,num2):
    if num1 == 1 and num2 == 1:
        return 1
    else:
        return num1 * num2 * recursive_function2(num1 - 1, num2 - 2)
```

(a) 1
(b) 48
(c) Infinite recursion
(d) 12
(e) 0