THE UNIVERSITY OF AUCKLAND

Semester Two, 2018
Campus: City

TEST

COMPUTER SCIENCE

Principles of Programming

(Time Allowed: 75 Minutes)

NOTE:
You must answer **all** questions in this test.
No calculators or smart watches are permitted.
Answer in the space provided in this booklet.
There is space at the back for answers which overflow the allotted space.

Surname	**POSSIBLE SOLUTIONS**
Forenames	
Preferred Name (if different to forenames)	
Student ID	
Username	
Lab Times	

Q1 (/20)	**Q4** (/20)
Q2 (/20)	**Q5** (/20)
Q3 (/20)	**TOTAL** (/100)

Question 1 (20 marks)

a) Complete the output produced by the following code.

```
result = 22 // 3 ** 2 - 1 + 12 / 5 // 2 * 3 - 1
print("result:", result)
```

```
result: 3.0
```

(3 marks)

b) Complete the output produced by the following code.

```
result1 = 17 % 7
result2 = 12 % 21
result3 = 24 % 3
print("result1:", result1, "result2:", result2,
                                        "result3:", result3)
```

```
result1: 3  result2:  12  result3: 0
```

(3 marks)

c) Complete the output produced by the following code.

```
num1 = 4
num2 = 3
num3 = num1
num2 = num2 + num3 * 2
num1 = num2 - num1
num2 = num2 + num3 * num3 - num1
print("Numbers:", num1, num2, num3)
```

```
Numbers: 7 20 4
```

(3 marks)

d) What is the smallest possible number and what is the largest possible number which can be printed by the following code?

```
import random

random1 = random.randrange(4, 16, 2)
random2 = random.randrange(1, 16, 3)
total = random1 + random2
print(total)
```

Smallest: 5 Largest: 27

(4 marks)

e) Complete the output produced by the following code.

```
phrase = "Enjoy today!"
bits_of_string = (phrase[-3] + phrase[-3:] + " - " +
                        phrase[2: 5] + " - " + phrase[9: 1: -3])
print("Output:", bits_of_string)
```

Output: aay! – joy - ato

(4 marks)

f) Given the following code, what is the type of each of the three Python objects: `object1`, `object2` and `object3`?

```
a_string = "Interesting"
number = 4532
a_list = [4, 7, "71"]

object1 = a_list[1] + number / 3
object2 = len(a_string * 4)
object3 = a_list[2] * 3
```

type of object1: float

type of object2: int

type of object3: str

(3 marks)

Question 2 (20 marks)

a) Assume that the variable, `words`, has been initialised to some string. Write the boolean expression which tests if the variable, `words`, has at least five characters and ends with the letter `"s"`.

```
len(words) >= 5 and words[-1] == "s"
```

(4 marks)

b) Assume that the variable, `value`, has been initialised to some integer. Write the boolean expression which tests if `value` is a two digit number and has a last digit (the right hand units digit) which is a 6.

```
len(str(value)) == 2 and value % 10 == 6
```

(4 marks)

c) Give the output produced when the following `main()` function is executed.

```
def main():
    function_ifs(70, 64)

def function_ifs(num1, num2):
    if num1 > num2 and num2 < 65:
        print("A", end = " ")
        if num2 % 5 == 0:
            print("B", end = " ")
        elif num2 % 2 == 0:
            print("C", end = " ")
        print("D", end = " ")
    else:
        if num2 > 100 or num1 < 65:
            print("E", end = " ")
        if num2 % 2 == 1:
            print("F", end = " ")
        print("G", end = " ")

    print("H", end = "")
```

```
A C D H
```

(4 marks)

d) How many times is the letter "A" printed by the following code?

```
for number in range(10, 50, 5):
    if number % 2 == 0:
        print("A")
    else:
        print("B")
```

```
4
```

(3 marks)

e) Complete the `while` loop in the following `print_number_of_guesses()` function. The function is passed a number (either 1, 2, 3, 4, 5 or 6) which the user has to guess. The function keeps prompting the user to guess the number until the user guesses the number correctly. The function then prints the number of guesses made by the user. For example, the function call `print_number_of_guesses(3)` executes as follows (the user input is shown in a larger bold font):

```
Enter number (1, 2, 3, 4, 5 or 6): 1
Enter number (1, 2, 3, 4, 5 or 6): 6
Enter number (1, 2, 3, 4, 5 or 6): 3
Number of guesses: 3
```

```
def print_number_of_guesses(secret_number):
    prompt = "Enter number (1, 2, 3, 4, 5 or 6): "
    number_of_guesses = 1
    user_number = int(input(prompt))
```

```
    while user_number != secret_number :
        number_of_guesses += 1
        user_number = int(input(prompt))

    print("Number of guesses:", number_of_guesses)
```

(5 marks)

Question 3 (20 marks)

a) Complete the `get_product_name()` function below which is passed a string parameter, the description of a product. The parameter string is made up of three parts: the product code, followed by a single `"-"`, followed by the product name, followed by a single `"-"` and finally the product price. The function returns the product name all in uppercase characters, i.e., middle part of the parameter string. For example, executing the following `main()` function with the completed `get_product_name()` function prints:

```
1. AMBRE ANTIQUE
2. WIND SONG
```

```
def main():
    result = get_product_name("bc13245-Ambre Antique-134.75")
    print("1.", result)
    print("2.", get_product_name("bc87966-Wind Song-18.20"))

def get_product_name(product):
```

```
    index1 = product.find("-") + 1
    index2 = product.rfind("-")
    product_string = product[index1: index2]

    return product_string.upper()
```

(7 marks)

b) Complete the `get_number()` function below which is passed three integer parameters: three single digit whole numbers between 1 and 9 inclusive. The function returns the whole number made up of the largest of the three parameter digits followed by the smallest of the three parameter digits (a number between 11 and 99 both inclusive). For example, executing the following `main()` function with the completed `get_number()` function prints:

```
1. 93
2. 61
```

```
def main():
    result = get_number(3, 7, 9)
    print("1.", result)
    print("2.", get_number(2, 6, 1))
def get_number(num1, num2, num3):
```

```
    largest = max(num1, num2, num3)
    smallest = min(num1, num2, num3)
    num_string = str(largest) + str(smallest)

    return int(num_string)
```

(7 marks)

c) The following function definition is very poorly named (the function is named `aaa`) and the two parameters are also very poorly named (the two parameters are named `b` and `c`). Rewrite the function header (**just the first line** of the function) using descriptive names for the function and the two parameters.

```
def aaa(b, c):
    length = len(b)
    num_stars = (c - length) // 2
    stars = "*" * num_stars
    centred_word = stars + b + stars
    return centred_word
```

```
def get_centred_word(word, total_length):
    ...
```

(6 marks)

Question 4 (20 marks)

a) In the boxes below, show each element of `list1` and `list2` after the following code has been executed. Use as many of the boxes as you need.

```
list1 = list(range(0, 35, 10))
list2 = list(range(20, -1, -4))
```

list1

0	10	20	30				
0	1	2	3	4	5	6	7

list2

20	16	12	8	4	0		
0	1	2	3	4	5	6	7

(6 marks)

b) Complete the output produced when the following `main()` function is executed.

```
def main():
    number_list = [9, 3, 6, 2, 8]
    result_list = invert(number_list)
    print("Result:", result_list)

def invert(numbers):
    result = []
    for num in numbers:
        result = result + [10 – num]
    return result
```

```
Result: [1, 7, 4, 8, 2]
```

(6 marks)

c) Complete the `get_converted_list()` function which takes a list of words (`word_list`) as a parameter. The function returns a **new list** of all the words from the parameter list altered in the following ways:

- the last letter of each word is removed from the end of the word and inserted at the front of the word,
- each word of the returned list is all in lower case characters.

For example, executing the following `main()` function using the completed `get_converted_list()` function gives the output:

```
['sjes', 'ncai', 'yamit', 'nraean']
```

Note: you can assume that every word in the parameter list contains at least one character.

```
def main():
    word_list = ["Jess", "Cain", "Amity", "Raeann"]
    words = get_converted_list(word_list)
    print(words)

def get_converted_list(word_list):
```

```
converted_list = []

for word in word_list:
    word2 = word[-1] + word[1: -1]
    word2 = word2.lower()

    converted_list.append(word2)


return converted_list
```

(8 marks)

Question 5 (20 marks)

a) Give the output produced by the following program.

```
def main():
    print("A", end = "")
    do1()

def do1():
    do2()
    print("B", end = "")

def do2():
    do3()
    print("C", end = "")
    do3()

def do3():
    print("D", end = "")

main()
```

ADCDB

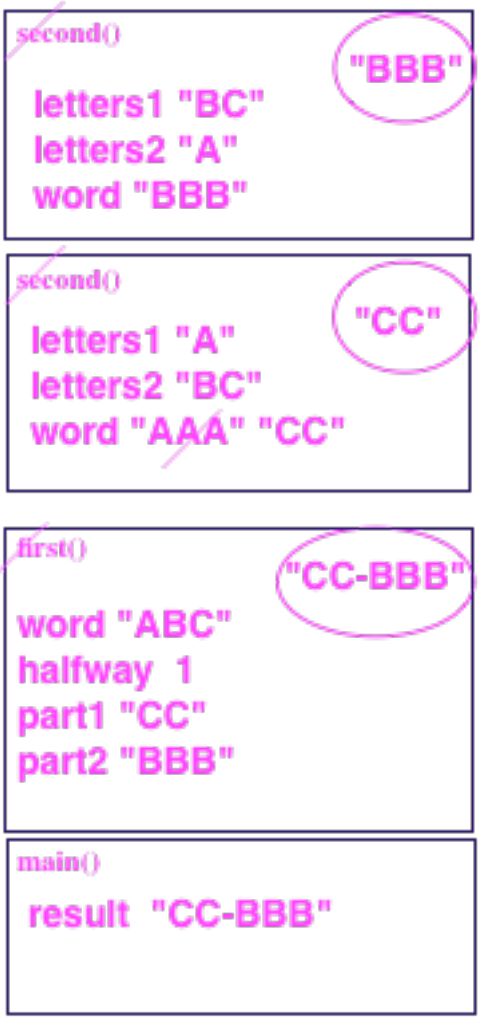
(6 marks)

b) Using the code trace technique taught in lectures, perform a code trace on the program on the next page and show the output.

```
def main():
    result = first("ABC")
    print(result)

def first(word):
    halfway = len(word) // 2
    part1 = second(word[:halfway], word[halfway:])
    part2 = second(word[halfway:], word[:halfway])
    return part1 + "-" + part2

def second(letters1, letters2):
    word = letters1[0] * 3
    print(word)
    if len(letters1) < len(letters2):
        word = letters2[-1] * 2
    return word

main()
```

The output:

AAA
BBB
CC-BBB



(14 marks)