THE UNIVERSITY OF AUCKLAND

Semester 2, 2019
Campus: City

TEST

COMPUTER SCIENCE

ANSWERS

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	
Forenames	
Preferred Name (if different to forenames)	
Student ID	
Username	
Lab Time	

Q1 (/25)	**Q4** (/15)	**TOTAL** (/100)
Q2 (/15)	**Q5** (/15)	
Q3 (/15)	**Q6** (/15)	

Question 1 (25 marks)

a) Complete the output produced by the following code.

```
num1 = 6
num2 = num1 + 10
num3 = num2 + 4
num1 = num1 + 3
num2 = num2 - num1
num3 = num3 + (num1 - num2)
print("num1:", num1, "num2:", num2, "num3:", num3)
```

```
num1:  9  num2:  7  num3: 22
```

(3 marks)

b) Complete the output produced by the following code.

```
num1 = 5 * (8 - 2 * 3) - 2 * 3 ** 2 // 4
num2 = 16 // 3 + 2 / 4 - 11 // 2
print("num1:", num1, "num2:", num2)
```

```
num1:  6  num2: 0.5
```

(4 marks)

c) Complete the output produced by the following code.

```
message = "PYTHON 3"
part1 = message[3] + message[5] + message[-1]
part2 = message[0] * (len(message) // 2)
print("part1:", part1, "part2:", part2)
```

```
part1:  HN3  part2: PPPP
```

(4 marks)

d) Complete the output produced by the following code.

```
message = "Python is amazing"
part1 = message[3: 11: 2]
part2 = message[-4: -8: -2]
print("part1:", part1, "part2:", part2)
```

part1: **hni** (space after the 'i') part2: **zm**

(4 marks)

e) Given the formula,

$$f(n) = \frac{(1+\sqrt{5})^n}{2}$$

complete the code which calculates and prints `f(n)` (`function_n`) to the nearest whole number where n is a positive whole number. Below are two example outputs from the completed program (using different values for the variable, `n`).

```
For n equal to 5 f(n) evaluates to 177
```

```
For n equal to 9 f(n) evaluates to 19459
```

```
import math

n = …     #assume that n has been assigned a whole number

function_n = (1 + math.sqrt(5)) ** n
function_n = function_n / 2


print("For n equal to", n, "f(n) evaluates to", round(function_n))
```

(5 marks)

f) A string describing a product item is made up of the description of the item, the item code within parentheses (round brackets) and the price. Three example item descriptions are:

```
"Classic kitchen Sink Bowl Top Mount, (28724), Price $697"
"Rhino Aluminium Multi-Purpose Ladder, (31374300), Price $248.18"
"Parkwood Cedar Entrance Door, (5307), Price $1783"
```

You can assume that the only parentheses (round brackets) in the item string are those surrounding the item code. Complete the following `main()` function which prints the item code. For example, if the variable, `item`, is assigned the string:

```
"Rhino Aluminium Multi-Purpose Ladder, (31374300), Price $248.18"
```

the output of the completed code is:

```
Item code: 31374300
```

```
import math

item = ... #assume that item has been assigned a product string


code_start_index = item.find("(") + 1
code_end_index = item.rfind(")")
code = item[code_start_index: code_end_index]


print("Item code:", code)
```

(5 marks)

Question 2 (15 marks)

Complete the following `main()` function which generates two random dice values (whole numbers between 1 and 6 both inclusive), prints the two dice values and calculates the biggest possible number which can be calculated from the two dice values. When calculating the biggest possible two digit number, each dice can only be used once. The final line of the program prints the biggest possible two digit number.

Below are two possible outputs obtained by executing the completed main() function:

```
Dice values: 2 5
Biggest possible number: 52
```

```
Dice values: 4 1
Biggest possible number: 41
```

```
Dice values: 5 5
Biggest possible number: 55
```

```
import random

def main():
    dice1 = random.randrange(1, 7)
    dice2 = random.randrange(1, 7)

    print("Dice values:", dice1, dice2)

    smallest = min(dice1, dice2)
    biggest = max(dice1, dice2)

    number = str(biggest) + str(smallest)

    print("Biggest possible number:", number)
```

(15 marks)

Question 3 (15 marks)

Part a) and Part b) of this question refer to the following function. You should assume that this function is defined in both the Part a) program and the Part b) program.

```
def mystery(letters1, letters2):
    shortest_len = min(len(letters1), len(letters2))
    index = 0
    result = ""
    while index < shortest_len:
        letter_to_test = letters1[index]
        if letter_to_test in letters2 and letter_to_test not in result:
            result = result  + letter_to_test
        index = index + 1

    return result
```

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

```
def main():
    word1 = "ABCA"
    word2 = "CATALOGUE"
    print("1:", mystery(word1, word2))
```

```
1:  AC
```

(4 marks)

b) In the `main()` function below, assign a string to the `word2` variable so that the output of the `main()` function is:

```
2: cs
```

```
def main():
    word1 = "cabs"

    word2 = "costly"    #word must include 'c' and 's' and not include
                        #'a' or 'b'.
    print("2:", mystery(word1, word2))
```

(4 marks)

c) Define the `get_digit_sum()` function which is passed a string parameter which is made up of EXACTLY two digits. The function returns the sum of the two digits which make up the parameter string. For example, when the following `main()` function is executed using the completed function, the output is:

```
1: 5
2: 11
```

```
def main():
    print("1:", get_digit_sum("23"))
    print("2:", get_digit_sum("92"))
```

```
def get_digit_sum(digits):

    number1 = int(digits[0])
    number2 = int(digits[1])

    return number1 + number2
```

(7 marks)

Question 4 (15 marks)

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

```
def main():
    function_if(5, 7, 7)

def function_if(num1, num2, num3):
    if num1 > num2 and num1 < num3:
        print("A", end = " ")
    elif num2 > num1 and num2 > num3:
        print("B", end = " ")
    else:
        print("C", end = " ")

    if num3 > num1 or num2 > num3:
        print("D", end = " ")
        if not num1 % 2 == 0 or num1 % 3 == 0:
            print("E", end = " ")
        else:
            print("F", end = " ")
        print("G", end = " ")
```

C D E G

(3 marks)

b) Complete the output produced by the following code.

```
value1 = 5
value2 = 7
word1 = "HOPPING"

result1 = value1 < value2 and not value2 > len(word1) or value1 == 5
result2 = not value1 < value2 or value2 > len(word1) and value1 == 5

print("result1:", result1, "result2:", result2)
```

result1: True result2: False

(4 marks)

c) Assume that the variables, `value1` and `value2` have both been assigned some integer value. Write a boolean expression which evaluates to `True` if `value1` is exactly divisible by `value2`. Otherwise the expression evaluates to `False`.

```
value1 % value2 == 0
```

(2 marks)

d) Assume that the variables, `word1` and `word2` have both been assigned some string. Write a boolean expression which evaluates to `True` if both `word1` and `word2` contain the lowercase letter `"a"`. Otherwise the expression evaluates to `False`.

```
"a" in word1 and "a" in word2
```

(2 marks)

e) Complete the output produced by the following code.

```
result1 = False
result2 = True
result3 = result1 and result2
result2 = not result2
result1 = not result3 or result2

print("result1:", result1, "result2:", result2)
```

```
result1:    True        result2: False
```

(4 marks)

Question 5 (15 marks)

a) Give the output produced by the following code.

```
number1 = 21
number2 = 13
while number1 > number2:
    print(number1, number2)
    number1 = number1 - 1
    number2 = number2 + 2
```

```
21 13
20 15
19 17
```

(4 marks)

b) Give the output produced by the following code.

```
special_letters = "ue"
phrase = "Super duper"
index = 0
while index < len(phrase):
    print(phrase[index], end="")
    if phrase[index] in special_letters:
        index = index + 1
    index = index + 1
```

```
Sue due
```

(4 marks)

c) In the program below, complete the `get_positive_even_number()` function. This function continuously prompts the user to enter an even number greater than zero until the user enters an even number which is greater than zero. The function returns the number entered by the user. Below are two possible outputs produced when the following `main()` function is executed using the completed `get_positive_even_number()` function (the user input is shown in bold).

```
Enter an even number greater than 0: 35
Enter an even number greater than 0: 53
Enter an even number greater than 0: 1
Enter an even number greater than 0: 28
User number: 28
```

```
Enter an even number greater than 0: -6
Enter an even number greater than 0: 58
User number: 58
```

```
def main():
    user_number = get_positive_even_number()
    print("User number:", user_number)

def get_positive_even_number():
```

```
number_prompt = "Enter an even number greater than 0: "
user_number = int(input(number_prompt))

while user_number % 2 == 1 or user_number <= 0:
    user_number = int(input(number_prompt))

return user_number
```

(7 marks)

Question 6 (15 marks)

Using the code trace technique taught in lectures, perform a code trace on the program below and show the output.

```
def main():
    phrase = "go for it"
    result1 = do_1(phrase)
    index =  phrase.find(" ")
    result2 = do_2(phrase, 0, index)
    print("A.", result1, result2)

def do_1(words):
    print("B.")
    position = words.rfind(" ")
    result1 = do_2(words, position + 1, len(words))
    print("C.", result1)
    return len(result1)

def do_2(phrase, index1, index2):
    print("D.", index1, index2)
    words = phrase[index1 :index2]
    return words

main()
```

Show the output:

B.
D. 7 9
C. it
D. 0 2
A. 2 go

do_2()
phrase "go for it" "go"
index1 0
index2 2
words "go"

do_2()
phrase "go for it" "it"
index1 7
index2 9
words "it"

do_1()
words "go for it" 2
position 6
result1 "it"

main()
phrase "go for it"
result1 2
index 2
result2 "go"