THE UNIVERSITY OF AUCKLAND

SECOND SEMESTER, 2015
Campus: City

COMPUTER SCIENCE

Principles of Programming

(Time Allowed: TWO hours)

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

<table>
<thead>
<tr>
<th>Surname</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>Forenames</td>
<td></td>
</tr>
<tr>
<td>Student ID</td>
<td></td>
</tr>
<tr>
<td>Login (UPI)</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Q1</th>
<th>Q4</th>
<th>Q7</th>
</tr>
</thead>
<tbody>
<tr>
<td>(/24)</td>
<td>(/7)</td>
<td>(/8)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Q2</th>
<th>Q5</th>
<th>Q8</th>
</tr>
</thead>
<tbody>
<tr>
<td>(/24)</td>
<td>(/9)</td>
<td>(/6)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Q3</th>
<th>Q6</th>
<th>TOTAL</th>
</tr>
</thead>
<tbody>
<tr>
<td>(/8)</td>
<td>(/14)</td>
<td>(/100)</td>
</tr>
</tbody>
</table>
Question 1 (24 marks)

a) Complete the output produced by the following code.

```python
result = 2 * 3 ** 2 - 3 // 2 * 2
print("Result:", result)
```

Result: 16

(2 marks)

b) Complete the output produced by the following code.

```python
result = str(2 * 3) * 4 + "s"
print("Result:", result)
```

Result: 6666s

(2 marks)

c) Complete the output produced by the following code.

```python
value1 = 24 % 10
value2 = 5 % 10
value3 = 10 % 10
print("value1:", value1, "value2:", value2, "value3:", value3)
```

value1: 4 value2: 5 value3: 0

(2 marks)
d) Complete the output produced by the following code.

```python
a = 4
b = a
c = b
a = a - 1
c = c + a
b = c + b
print("a:", a, "b:", b, "c:", c)
```

```
a: 3  b: 11  c: 7
```

(2 marks)

e) Complete the output produced by the following code.

```python
saying = "C'est la vie!"
result = saying.find("st") + saying.rfind("al")
print("Result: ", result)
```

```
Result: 2
```

(2 marks)

f) Complete the output produced by the following code.

```python
saying = "Leave room for magic"
say_little = saying[7: 9] + saying[-3:]
print("say_little:", say_little)
```

```
say_little: oogic
```

(2 marks)
g) Complete the output produced by the following code if the user enters 5 at the prompt.

```python
user_input = input("Enter number: ")
result = int(user_input * 3) + 1
print("Result: ", result)
```

Enter number: 5
Result: 556

(2 marks)

h) What is the largest possible number which can be printed by the following code? You can assume that the random module has been imported.

```python
value = random.randrange(3, 20, 3)
print(value)
```

18

(2 marks)

i) Given the following code (the right hand side of the first two statements is not shown but you can assume the code executes without error):

```python
num1 = ...
num2 = ...
value1 = not (num1 > num2)
value2 = 34 / 2 + 1
```

what is the type of the two Python objects `value1` and `value2`?

- `value1` is an object of type: `bool`
- `value2` is an object of type: `float`

(2 marks)
j) Give the output produced by the following code.

```python
sentence = "Life is understood backwards, but lived forwards."
words = sentence.split()
print(words[0], words[-2])
```

```
Life lived
```

(2 marks)

k) Given the following function:

```python
def fiddle1(a_list):
    for index in range(len(a_list)):
        word = a_list[index]
        word = word + "s"
```

complete the output produced by the following section of code:

```python
list1 = ["one ", "two"]
fiddle1(list1)
print("list1: ", list1)
```

```
list1: [ 'one', 'two' ]
```

(2 marks)
l) Given the following function:

```python
def fiddle2(list1, list2):
    position = 1
    for number in list2:
        if position < len(list1):
            list1[position] = list1[position] + number
            position = position + 1
```

complete the output produced by the following section of code:

```python
list1 = [4, 2, 5]
list2 = [1, 3, 5, 6]
fiddle2(list1, list2)
print("list1:", list1)
```

```
list1: [ 4, 3, 8 ]
```

(2 marks)
Question 2 (24 marks)

a) Complete the for...in loop so that the output is:

12 18 24 30

```python
for value in range(4, 11, 2):
    num = value * 3
    print(num, end = " ")
```

(3 marks)

b) Complete the for...in loop so that the output is:

35 30 25 20 15 10 5 0

```python
for num in range(35, -1, -5):
    print(num, end = " ")
```

(3 marks)

c) Complete the following if statement which prints "YES" if the parameter, num, is either a positive even number or a negative odd number. Consider zero to be a positive number.

```python
def print_result(num):
    if (num >= 0 and num % 2 == 0) or (num < 0 and num % 2 == 1):
        print("YES")
```

(3 marks)
Part d) and Part e) refer to the following function:

```python
def if_function(num1, num2, num3):
    if num1 < num2 and num3 > num2:
        print("A")
        if num1 + num2 > 100:
            print("B")
        print("C")
    elif num3 < num1:
        if num1 + num2 > 200:
            print("D")
        else:
            print("E")
        print("F")
    else:
        if num2 > num1:
            print("G")
        elif num3 < num2:
            print("H")
        else:
            print("I")
    print("J")
```

d) Give the output produced by the following function call:

```python
if_function(32, 150, 100)
```
c) Give the output produced by the following function call:

```python
if_function(10, 200, 5)
```
g) Complete the following code which prints a random name from the list, names_list. Note that the full list is not shown here. You can assume that the random module has been imported.

```python
names_list = ['Dorothy', 'Thomas', ...
                  'Susan', 'Joseph']

random_name_position = random.randrange(0, len(list_of_names))

random_name = list_of_names[random_name_position]

print(random_name)
```

(3 marks)

h) Give the output produced by the following code:

```python
previous = 14
num = previous + 10

while num < 50:
    digits = str(previous) + str(num)
    print(digits, end = " ")
    previous = num
    num = num + 10
```

```
1424 2434 3444
```

(3 marks)
**Question 3 (8 marks)**

Part a) and Part b) below refer to the following function:

```python
def is_a_mystery(word1, word2, letter1, letter2):
    pos1 = word1.find(letter1)
    pos2 = word2.find(letter2)
    if pos1 == -1 or pos2 == -1:
        return "A"
    elif pos1 > pos2:
        return "B"
    return "C"
```

a) Complete the following code so that the output produced is "A".

```python
letter = is_a_mystery("welcome", "valid", 'x', "d")
print(letter)
```

(2 marks)

b) Complete the following code so that the output produced is "B".

```python
letter = is_a_mystery("welcome", "valid", 'm', "a")
print(letter)
```

(2 marks)
c) Complete the `get_new_text()` function which is passed three parameters:

- `text` - the original text
- `to_add` - the text to be inserted into the original text
- `position` - the position in the original text before which the text is to be inserted.

The function returns the new string containing the inserted text, e.g., the following code:

```python
print(get_new_text(get_new_text("1234", "abc", 0)))
print(get_new_text("I feel marvellous", "always ", 2))
print(get_new_text("Rose", "mary", 4))
```

prints:

```
abc1234
I always feel marvellous
Rosemary
```

```python
def get_new_text(text, to_add, position):
    part1 = text[0: position]
    part2 = text[position:]
    return part1 + to_add + part2
```

(4 marks)
**Question 4 (7 marks)**

a) Complete the output produced by the following program.

```python
def process_dict(list1, list2):
    a_dict = {}

    for index in range(len(list1)):
        key = list1[index]
        if key in a_dict:
            a_dict[key].append(list2[index])
        else:
            a_dict[key] = [list2[index]]

    return a_dict

def main():

    a_dict = process_dict(names, grades)

    print("Ava:", a_dict["Ava"])
    print("Lee:", a_dict["Lee"])

main()

Ava:  ['A', 'B']
Lee:  ['B', 'C', 'B']
```

(3 marks)
b) Complete the `print_if_over_bar()` function which is passed two parameters:

```
    a_dict       - a Python dict object
    bar          - a number
```

The function prints all the key:value pairs in the dictionary which have a value greater than the `bar` parameter, e.g., the following code:

```
    a_dict = {"0034":9, "0056":32, "0045":17, "0033":21}
    print_if_over_bar(a_dict, 15)
    print()
    print_if_over_bar(a_dict, 20)
```

prints:

```
0056  32
0045  17
0033  21

0056  32
0033  21
```

```python
def print_if_over_bar(a_dict, bar):
    for key in a_dict:
        if a_dict[key] > bar:
            print(key, a_dict[key])
```

(4 marks)
Question 5 (9 marks)

a) Complete the output produced by the following program:

```python
def unknown(words):
    count = 0
    for word in words:
        if len(word) > 1 and word[0] == word[-1]:
            count += 1
    return count

def main():
    print("1:", unknown(['aba', 'xyz', '123', 'bbb']))
    print("2:", unknown(['qpq', 'xy', 'xyx', 'xx']))
    print("3:", unknown(['aaa', 'e', 'abc', 'hello']))
    print("4:", unknown(['', '123', 'e', 'hello']))
```

```
1: 2
2: 3
3: 1
4: 0
```

(4 marks)
b) Complete the `get_even()` function which takes a tuple as a parameter and returns a new tuple containing all the even numbers from the parameter tuple. For example, the following code:

```python
new_tuple = get_even((1, 8, 6, 12, 4, 67))
print(new_tuple)
```

prints: `(8, 6, 12, 4)`

```python
def get_even(a_tuple):
    new_list = []
    for num in a_tuple:
        if num % 2 == 0:
            new_list.append(num)
    return tuple(new_list)
```

(5 marks)
Question 6 (14 marks)

a) Complete the output produced by the following code.

```python
list1 = [1, 2]
list2 = list1 * 2
list3 = list1 + list2
list4 = list1 + [2]
list5 = list2[2: len(list2)]
list6 = list2[2: 0: -1]

print("1:", list1)
print("2:", list2)
print("3:", list3)
print("4:", list4)
print("5:", list5)
print("6:", list6)
print("7:", list1 == list5)
print("8:", list1 is list5)
print("9:", list1 == list6)
print("10:", list2 == list4)
```

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>[1, 2]</td>
</tr>
<tr>
<td>2</td>
<td>[1, 2, 1, 2]</td>
</tr>
<tr>
<td>3</td>
<td>[1, 2, 1, 2, 1, 2]</td>
</tr>
<tr>
<td>4</td>
<td>[1, 2, 2]</td>
</tr>
<tr>
<td>5</td>
<td>[1, 2]</td>
</tr>
<tr>
<td>6</td>
<td>[1, 2]</td>
</tr>
<tr>
<td>7</td>
<td>True</td>
</tr>
<tr>
<td>8</td>
<td>False</td>
</tr>
<tr>
<td>9</td>
<td>True</td>
</tr>
<tr>
<td>10</td>
<td>False</td>
</tr>
</tbody>
</table>
b) Complete the output produced by the following code.

```python
list1 = [1, 2, 3, 4, 5]
list1.append(100)
print("1:", list1)

list1.insert(2, 6)

print("2:", list1)
print("3:", list1.pop(0))
print("4:", list1)
```

1: [1, 2, 3, 4, 5, 100]  
2: [1, 2, 6, 3, 4, 5, 100]  
3: 1  
4: [2, 6, 3, 4, 5, 100]

(4 marks)
c) Complete the `reverse_append_pop()` function which takes a list as a parameter and returns a new list containing the elements from the parameter list in reverse order. Your function must **ONLY USE** the Python list `append()` and the Python list `pop()` methods.

Your function should continue removing the last element of the parameter list and appending it to the end of the new list until the parameter list is empty. For example, the following code:

```python
new_list = reverse_append_pop([1, 2, 3, 4, 5])
print(new_list)
```

prints: `[5, 4, 3, 2, 1]`

def reverse_append_pop(a_list):

```python
new_list = []
while len(a_list) > 0:
    new_list.append(a_list.pop())
return new_list
```

(5 marks)
**Question 7 (8 marks)**

Complete the `process_marks()` function which is passed three parameters:

- `infilename` - the name of a text file containing a list of marks (one mark per student) which is to be read into the program,
- `outfilename` - the name of the output file where the summary is to be written,
- `course_name` - the name of the course.

The input file contains a list of marks (whole numbers), one mark per line. The function reads the list of marks from the input file, processes the marks by calculating the total number of marks (the number of students in the class) and the average mark, and finally, writes a summary to the outfile e.g., if the file, "Marks565.txt", is the file shown on the right, the following code:

```
process_marks("Marks565.txt", "Summary561.txt", "CompSci 565")
```

produces the file:

```
CompSci 565
Number of students: 6
Average mark: 66
```

The summary is made up of three lines of text: the course name, the number of students preceded by the string "Number of students: ", and the third line is the average mark which is rounded to the closest integer value preceded by the string "Average mark: ".

```python
def process_marks(infilename, outfilename, course_name):
    infile = open(infilename, "r")
    outfile = open(outfilename, "w")
    contents = infile.read()
    infile.close()
    marks_list = contents.split()
    number_of_students = len(marks_list)
    total = 0

    for mark in marks_list:
        total = total + int(mark)

    average = round(total / number_of_students)
    outfile.write(course_name + "\n")

    outfile.write("Number of students: " + str(number_of_students) + "\n")
    outfile.write("Average mark: " + str(average) + "\n")
```

```
outfile.write("Average mark: 
str(average) + "\n")
outfile.close()
Question 8 (6 marks)

Parts a) and b) of this question refer to the following program.

```python
from tkinter import *

def draw_shapes(a_canvas):
    size = 10
    top = size
    left_hand_side = size

    for line_num in range(0, 3):
        left = left_hand_side
        if line_num == 0:
            show_circle = True
            show_rect = False
        elif line_num == 1:
            show_circle = True
            show_rect = True
        else:
            show_circle = False
            show_rect = True

        for shape_num in range(0, 6):
            rect = (left, top, left + size, top + size)
        if shape_num % 2 == 0:
            if show_rect:
                a_canvas.create_rectangle(rect)
            else:
                if show_circle:
                    a_canvas.create_oval(rect)

        left = left + size
        top = top + size

main()
```

```python
def main():
    root = Tk()
    root.title("A Canvas")
    root.geometry("100x60+10+20")
    a_canvas = Canvas(root)
    a_canvas.pack(fill=BOTH, expand = True)
    draw_shapes(a_canvas)
    root.mainloop()

main()
```
a) In total how many times is the statement marked Position A in the program above executed when the program executes?

18

(2 marks)

b) As accurately as possible, in the window below, show what is drawn by the above program. Grid lines have been drawn in the window to help you. The gap between adjacent gridlines is 10 pixels.

(4 marks)