THE UNIVERSITY OF AUCKLAND

SUMMER SEMESTER, 2015
Campus: City

COMPUTER SCIENCE

TEST SOLUTIONS

Principles of Programming

(Time Allowed: 75 minutes)

NOTE:
You must answer **all** questions in this test.
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.

Surname	
Forenames	
Student ID	
Login (UPI)	
Lab Time	

Q1 (/12)	**Q4** (/20)	
Q2 (/14)	**Q5** (/18)	**TOTAL** (/100)
Q3 (/24)	**Q6** (/12)	

ID: .....

Question 1 (12 marks)

a) Complete the output produced by the following code.

```
num1 = 5
num2 = 4

result = (num2 ** 2 - num1) // 2 / 2 - 1
print("Result:", result)
```

```
Result: 1.5
```

(2 marks)

b) Complete the output produced by the following code.

```
num1 = 5
num2 = 3
result = str(num1) * (num2 - 1)  + "0"
print("Result:", result)
```

```
Result: 550
```

(2 marks)

c) Complete the following statement which assigns a random number which is either 8, 10 or 12 to the variable, `number`. You can assume that the `random` module has been imported.

```
number =  random.randrange(8, 13, 2)
```

(2 marks)

CONTINUED

d) Complete the output produced by the following code.

```
message = "Be a voice not an echo!"

another_message = message[5: 9] + message[-3]
print("Letters:", another_message)
```

```
Letters: voich
```

(2 marks)

e) Assume that the variables `num1` and `num2` have been initialised. Write a Python boolean expression which evaluates to `True` if `num2` is an even number and `num1` is greater than `num2`.

```
num2 % 2 == 0 and num1 > num2
```

(2 marks)

f) Assume that the variables `num1` and `num2` have been initialised. Write a Python boolean expression which evaluates to `True` if the absolute difference between `num1` and `num2` is less than 3.

```
abs(num1 - num2) < 3
```

(2 marks)

Question 2 (14 marks)

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

```
def function_ifs(a, b, c):
    if a > b and a > c:
        print("A")
    elif not (a > b and a > c):
        print("B")
    else:
        print("C")

    if a > b or a > c:
        print("D")
    elif not(a > b or a > c):
        print("E")

    print("F")
```

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

```
function_ifs(4, 6, 8)
```

B
E
F

(4 marks)

ID: .....

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

```
function_ifs(8, 6, 4)
```

A
D
F

(4 marks)

c) This question uses the following function:

```
def mystery1(value1, value2):
    mix_up = value1[0] * value2
    return mix_up
```

Complete the call to the `mystery1()` function so that the output produced by the following code is:

```
CCCCC
```

Any word starting with the letter, "C".

```
letters = mystery1( "Cat", 5 )
print(letters)
```

(3 marks)

d) This question uses the following function:

```
def mystery2(word):
    length = len(word)
    if length > 3:
        letter1 = word[2]
        letter2 = word[0]
        return letter1 + "*" + letter2 + word[3:]

    return word + word
```

Complete the call to the `mystery2()` function so that the output produced by the following code is:

```
K*LE
```

Any four letter word starting with the letter, "L" and the third letter and fourth letters, "KE".

```
letters = mystery2( "LIKE" )
print(letters)
```

(3 marks)

ID: .....

Question 3 (24 marks)

a) Complete the `print_24_hour_time()` function which is passed an integer parameter, `minutes`. The function converts the number of minutes into 24 hour time. The function prints the hour followed by a ":" followed by the minutes.

For example, the following code:

```
print_24_hour_time(67)
print_24_hour_time(1316)
print_24_hour_time(4614)
```

prints:

```
1:7
21:56
4:54
```

```
def print_24_hour_time(minutes):
```

```
hours = minutes // 60
hours = hours % 24
mins = minutes % 60
print(str(hours) + ":" + str(mins))
```

(8 marks)

ID: .....

b) Complete the `remove_from_end()` function which is passed two parameters, a string and an integer. The function removes the number of characters (given by the second parameter) from the end of the string and returns the resulting string. If the number of characters in the string is less than the number of characters to remove, the function returns the string unchanged. For example, the following code:

```
phrase1 = remove_from_end("Cut_corners", 3)
phrase2 = remove_from_end("Last_straw", 6)
phrase3 = remove_from_end("hot potato", 3)
phrase4 = remove_from_end("Cat", 5)
print(phrase1, phrase2, phrase3, phrase4)
```

prints:

```
Cut_corn Last hot pot Cat
```

```
def remove_from_end(phrase, num_to_remove):
```

```
length = len(phrase)
if length < num_to_remove:
    return phrase

pos = length - num_to_remove
return phrase[0: pos]
```

(8 marks)

ID: .....

c) Complete the `get_little_name()` function which is passed a string parameter, `name`. The function returns a string made up of the first two letters of the name repeated. The function always returns a four character string in which the first letter is an uppercase letter and the last three letters are lowercase characters. You can assume that the parameter string has a length of two or more characters, that the first character is an uppercase character and that the second character is a lowercase character. For example, the following code:

```
name1 = get_little_name("Zachariah")
name2 = get_little_name("Sigourney")
name3 = get_little_name("Bettina")
name4 = get_little_name("Gilbert")
print(name1, name2, name3, name4)
```

prints:

```
Zaza Sisi Bebe Gigi
```

```
def get_little_name(name):
```

```
little_name = name[0:2]
little_name += little_name.lower()
return little_name
```

(8 marks)

ID: .....

Question 4 (20 marks)

a) What is the output produced by the following section of code?

```
num = 17
while num > 2:
    num = num - 3
    print(num)
    num = num – 2
```

```
14
9
4
```

(5 marks)

b) What is the output produced by the following section of code?

```
number = 60
counter = 5
for i in range(3, 16, 4):
    number = number - counter
    print(number)
```

```
55
50
45
40
```

(5 marks)

ID: .....

c) Convert the following code which uses a `for ... in` loop into equivalent code which uses a `while` loop.

```
total = 100

for number in range(10, 1, -4):
    print(number)
    total = total - number
```

```
number = 10
total = 100
while number > 1:
    print(number)
    total = total - number
    number = number - 4
```

(5 marks)

d) Using the `range()` function, complete the `for ... in` loop below so that the output, when the code is executed, is:

```
60   55   50   45   40   35   30   25
```

```
for number in range(60, 20, -5):

    print(number, "  ", end = "  ")

print()
```

(5 marks)

Question 5 (18 marks)

a) Complete the output produced by the following code:

```
a_list = [4, 2, 1, 3, 0, 5]
a_list[3] = a_list[3] - a_list[1]
a_list[1] = a_list[2] * a_list[5]
print("a_list:", a_list)
```

```
a_list: [4, 5, 1, 1, 0, 5]
```

(4 marks)

b) Complete the output produced by the following code:

```
a_list = [4, 2, 15]
a_list[0] = a_list[1] * 2
a_list = a_list + [2]
a_list = a_list + [a_list[1]]
print("a_list:", a_list)
```

```
a_list: [4, 2, 15, 2, 2]
```

(4 marks)

ID: .....

a) Complete the `get_evens_list()` function, which returns a new list which contains only the even numbers from the parameter list. For example, executing the following code using the completed function:

```
a_list1 = [4, 2, 1, 3, 6, 5]

a_list2 = get_evens_list(a_list1)

print("a_list2:", a_list2)
```

gives the output:

```
a_list2: [4, 2, 6]
```

```
def get_evens_list(a_list):
```

```
list2 = []
for item in a_list:
    if item % 2 == 0:
        list2 += [item]

return list2
```

(10 marks)

ID: .....

Question 6 (12 marks)

a) Using the code tracing technique shown in lectures, perform a code trace for the following program and give the output. Give the output in the space below and **show the code trace in the space provided on the next page**.

```
def main():
    print("A")
    function1()
    print("B")

def function1():
    print("C")
    first = function2(23)
    print("D", first)
    print("E")

def function2(value):
    num_str = "**" + str(value)
    pos = len(num_str)
    pos = pos - 3
    print("F", pos)
    return num_str[pos:]

main()
```

Give the output:

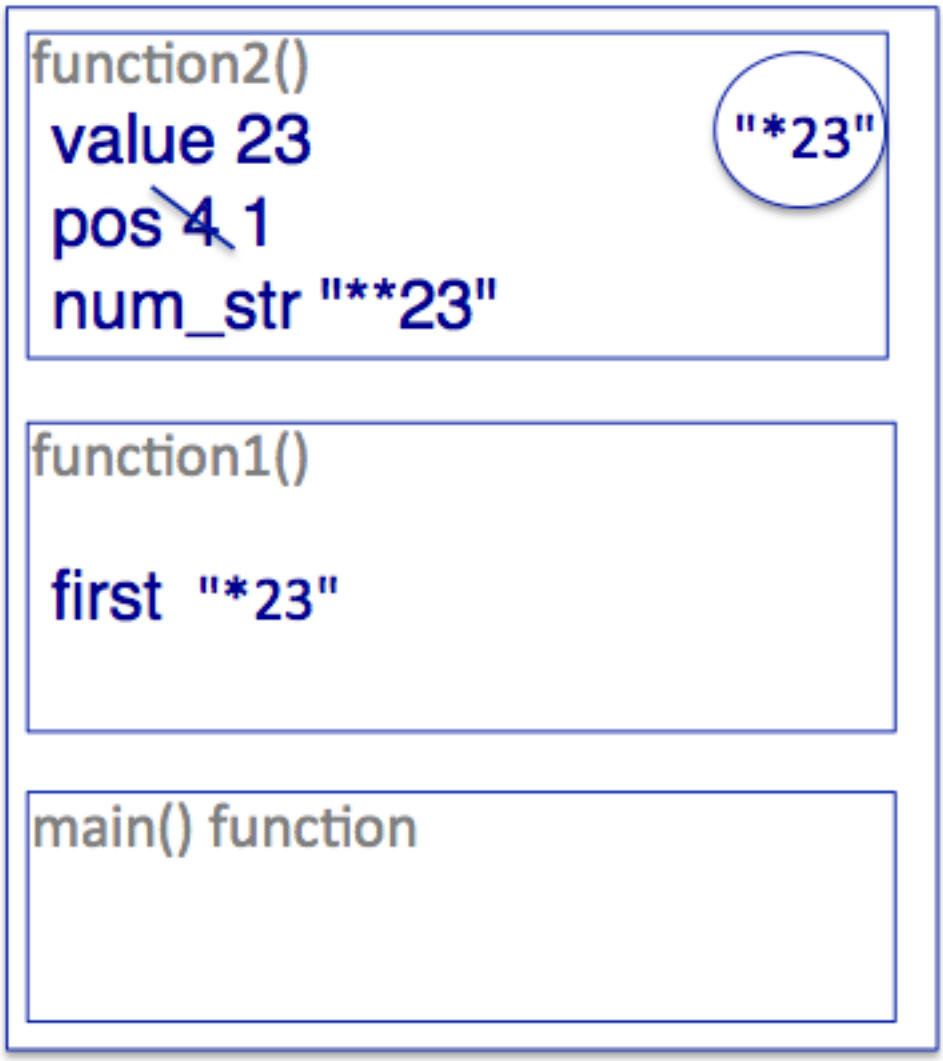
```
A
C
F 1
D *23
E
B
```

(6 marks)

CONTINUED

ID: .....

Show the code trace in the space below:



(6 marks)