THE UNIVERSITY OF AUCKLAND

Summer Semester, 2017
Campus: City

TEST

COMPUTER SCIENCE SOLUTIONS

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 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.

```
a = 6
b = 4
a = b
b = a
a = a + 3
b = b + 2

print("a:", a, "b:", b)
```

```
a:  7  b:  6
```

(3 marks)

b) Complete the output produced by the following code.

```
result = 5 + 3 * 3 ** 2 + 3 - 2 * 3.0
print("Result:", result)
```

```
Result: 29.0
```

(2 marks)

c) Complete the output produced by the following code.

```
value = 12
a = value % 5
b = 5 % value

print("a:", a, "b:", b)
```

```
a:  2  b:  5
```

(2 marks)

d) Give the output produced by the following code.

```
word = "BEDRAGGLED"
print(word[4], "-", word[-3:], "-", word[:2], "-", word[-2])
```

```
A - LED - BE - E
```

(3 marks)

e) Assume that the variable, `value`, has been initialised to some integer number. Write a boolean expression which tests if `value` is an even number greater than 7.

```
value % 2 == 0 and value > 7
```

(2 marks)

f) Complete the following program which first prints:

```
The amount to be saved $200
```

The program then prompts the user for the amount which they can save per week using the prompt, `"Savings per week $"`, calculates the number of weeks it will take to save this amount and prints the number of weeks. Assume the user always enters a whole number when prompted.

Note that the number of weeks is always a whole number, e.g., if the user says they can save $50 per week then it will take 4 whole weeks to save $200, if the user says they can save $60 per week it will still take 4 whole weeks to save $200 (they will only save $180 in 3 weeks). For example, the following screenshots show two different executions using the completed program:

```
The amount to be saved $200
Savings per week $22
The number of weeks needed is 10
```

The user input is shown in bold here and in a larger font.

```
The amount to be saved $200
Savings per week $60
The number of weeks needed is 4
```

```
amount = 200
prompt =  "Savings per week $"
print("The amount to be saved $", amount, sep="")


amount_per_week = int(input(prompt))
number_of_weeks = amount_needed // amount_per_week
if number_of_weeks * amount_per_week < amount_needed:
   number_of_weeks = number_of_weeks + 1

print("The number of weeks needed is",  number_of_weeks)
```

(8 marks)

Question 2 (20 marks)

a) Give the output produced by the following code.

```
previous = 3
number = 6
for num in range(number, 12, 3):
    print(previous, num)
    previous = num

print("end:", previous)
```

```
3  6
6  9
end: 9
```

(5 marks)

b) Give the output produced by the following code.

```
word = "EFFERVESCENT"
new_word = ""

for index in range(3, len(word), 4):
    letters = word[index - 1] + word[index]
    new_word = new_word + letters
    print(new_word)
```

```
FE
FEES
FEESNT
```

(5 marks)

c) Give the smallest possible number which can be printed by the program below and the largest possible number which can be printed.

```
import random
def main():
    random1 = random.randrange(4, 9)
    random2 = random.randrange(10, 25, 2)
    total = random1 + random2
    print(total)

main()
```

```
The smallest possible number: 14

The largest possible number: 32
```

(4 marks)

d) Rewrite the following code using an equivalent `while` loop instead of the `for` loop:

```
num = 3
for count in range(9, -4, -3):
    print(num)
    num = num + count

print("End")
```

```
num = 3
count = 9

while count > -4:
   print(num)
   num = num + count
   count = count - 3

print("End")
```

(6 marks)

Question 3 (20 marks)

a) Give the output produced by the following code.

```
number1 = 15
number2 = 11

if number1 > number2 and number2 < 11:
    print("A")
elif not number2 >= 11 or number1 > 20:
    print("B")
else:
    print("D")
    if number1 - number2 < 4:
        print("C")
print("E")
```

```
D
E
```

(5 marks)

b) Give the output produced by the following program.

```
def print_price_extras(num_days, concession_applies):
    per_day = 10
    price = num_days * per_day

    if num_days > 5:
        num_days =  5 + (num_days - 5) // 2
        price = num_days * per_day

    if concession_applies:
        price = price - price // 5
        print("Concession Price $", price, sep="")
    else:
        print("Price $", price, sep="")

def main():
    print_price_extras(5, True)
    print_price_extras(10, False)

main()
```

```
Price with concession $40
Price $70
```

(8 marks)

c) Complete the `get_changed_word()` function which is passed three parameters, a word, an index number and a single letter. The function replaces the character in the word at the position given by the index number with the single letter and returns the new word which is formed. You can assume that the `word` parameter always has a length which is greater than the value of the `index` parameter. For example, executing the following program with the completed function gives the output:

1. code -> cone
2. winner -> winter

```
def main():
    word1 = 'code'
    word2 = get_changed_word(word1, 2, 'n')
    print("   1.", word1, '->', word2)
    word1 = 'winner'
    word2 = get_changed_word(word1, 3, 't')
    print("   2.", word1, '->', word2)
```

```
def get_changed_word(word, index,
                                    new_letter):
  part1 = word[ :index]
  part2 = word[index + 1:]
  return part1 + new_letter + part2
```

(7 marks)

```
main():
```

Question 4 (20 marks)

a) Given the following code:

```
a_list = [1, 'two', 3, 'four']

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

what is the type of each of the three Python objects: `object1`, `object2` and `object3`?

```
object1 is of type: float

object2 is of type: str

object3 is of type: bool
```

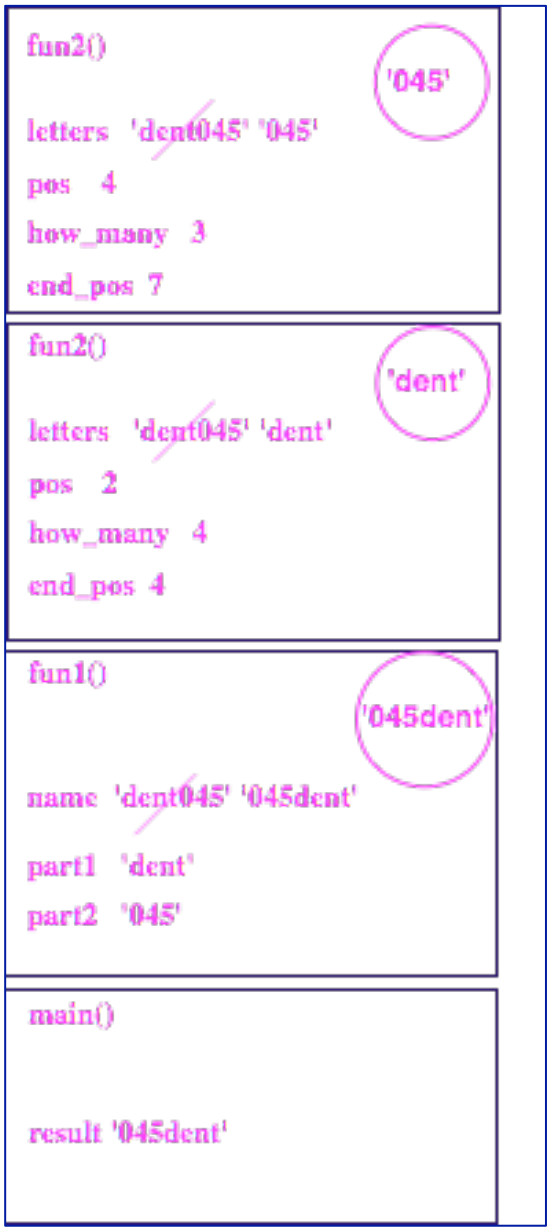
(6 marks)

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

```
def fun1(name):
    print("A.")
    part1 = fun2(name, 0, 4)
    part2 = fun2(name, 4, 3)
    print("B.")
    name = part2 + part1
    return name

def fun2(letters, pos, how_many):
    end_pos = pos + how_many
    letters = letters[pos: end_pos]
    print("C.", letters)
    return letters

def main():
    result = fun1("dent045")
    print("D.", result)

main()
```

Output

```
A.
C. dent
C. 045
B.
D. 045dent
```

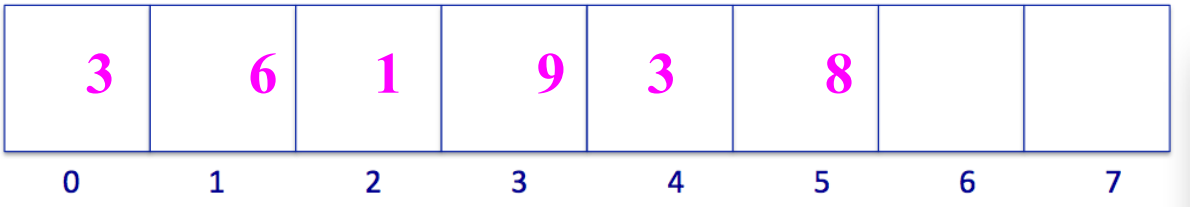


(14 marks)

Question 5 (20 marks)

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

```
a_list = [5, 1, 4, 3, 2]

a_list[0] = a_list[2] + a_list[4]
a_list[4] = a_list[4] * a_list[2]
a_list[-3] = a_list[-2] * 3
a_list = [a_list[1] + 2] + a_list
```



(6 marks)

b) Give the output produced by the following code:

```
a_list = [4, 2, 3, 7, 6, 5]
value = 10
amount = 0
for num in a_list:
    if value % num == 0:
        print(num)
        amount = amount + num
print("Finally:", amount)
```

```
2
5
Finally: 7
```

(6 marks)

c) Complete the `get_count()` function which is passed a list of names as a parameter. The function returns the number of names in the `names_list` parameter which have a length greater than 3 and end with a vowel. For example, executing the following program with the completed function prints:

```
Count: 3
```

Note: you can assume that all the elements in the list contain at least one character.

```
def main():
    names = ["Lee", "Vinnie", "Leila", "Tao", "Zaynab", "Maia"]

    count = get_count(names)
    print("Count:", count)


def get_count(names_list):
```

```
vowels = "aeiouAEIOU"


def get_count(words_list):
  vowels = "aeiouAEIOU"
  count = 0
  for word in words_list:
     last_letter = word[-1]
     if last_letter in vowels and
                                 len(word) > 3:
         count = count + 1

  return count
```

(8 marks)

```
main()
```