Name:

____

____

Student Number:

____

Please fill in your Student Number and Name.

Student Number : ____________________________________

University of Cape Town ~ Department of Computer Science

Computer Science 1015F ~ 2014

June Examination

*** SOLUTIONS ***

Question	Max	Internal	External
1	10		
2	20		
3	10		
4	20		
5	10		
TOTAL	**70**		

Marks : 70

Time : 120 minutes

Instructions:

a) Answer all questions.

b) Write your answers in PEN in the spaces provided.

c) Show all calculations where applicable.

Question 1 – Multiple choice [10]

Examine the Q1.py modules listed on the last sheet of the examination and answer the questions that follow. For each question, **write down ONE letter** corresponding to the correct answer.

(a) In **Q1.py**, `item` is an example of:

a. a function parameter

b. a data type

c. a string

d. a function

e. a literal

A

(b) An example of an **operator** from **Q1.py** is:

a. `*`

b. `or`

c. `//`

d. `==`

e. all of the above

E

(c) A **reserved word** used in **Q1.py** is:

a. `print`

b. `eval`

c. `int`

d. `elif`

e. all of the above

D

(d) What is the output of this Python3 module?

```
import Q1
print(Q1.swell(2))
```

a. `builtins.IndexError`

b. `44`

c. `8`

d. `4.0`

e. `22`

D

(e) What is the output of this Python3 module?

```
import Q1
print(Q1.swell("4.0"))
```

a. `builtins.TypeError`

b. nothing

c. `8`

d. `16`

e. `4.04.0`

E

(f) After importing **Q1.py**, which line of code will produce this exact output ?

```
1010
```

a. `Q1.chop(2020,2)`

b. `Q1.chop("10101010",5)`

c. `Q1.swell("505")`

d. A and B

e. B and C

A

(g) What is the output of this Python3 module?

```
import Q1
x=101
Q1.swell(x)
 print(x)
```

a. `builtins.TypeError`

b. nothing

c. `202`

d. `101`

e. `404.0`

D

(h) What is the output of this Python3 module?

```
import Q1
Q1.chop(Q1.swell("110.0"),-1)
```

a. `builtins.IndexError`

b. `builtins.TypeError`

c. `-440`

d. `110.0110.`

e. `20`

D

(i) What is the output of this Python3 module?

```
import Q1
Q1.chop("50.0",2)
```

a. `builtins.TypeError`

b. nothing

c. `50`

d. `0`

e. `25.0`

A

(j) Charles Babbage is famous for his design of:

a. The Vaucanson Duck

b. The Difference Engine

c. The Integrated Circuit

d. The Python Programming Language

B

Question 2 [20]

Examine the Q2.py module listed at the end of this examination and answer the following questions.

(a) Python is a dynamically typed language. Explain what this means, **using an example** from the `Q2.py` module. [3]

In a dynamically-typed language, variables can change type.[1] If the function ana(a,b) is sent non-strings as parameters a and b, they are changed to strings.[2]

(b) Explain clearly in general terms and in English (no code) what the `ana()` function does. [2]

Returns true if the words a and b are anagrams, false otherwise. [2]

(c) Explain clearly how the output of the `ana()` function will change with different inputs if the word **break** is changed to the word **continue**. [3]

This will create an infinite loop [1] when a and b are the same length, but not anagrams. [1] There will be no output in this case, it will loop forever. Other output will remain the same.[1]

(d) Write the code for the function `rmvVowels(worda)` so that it works as follows. This function should **return a string** corresponding to the characters in `worda` with all vowels removed. If `worda` is not a string, the function should return the empty string. For example (in the Python3 interpreter):

```
>>>rmvVowels("alphabet")
>>>'lphbt'
>>>rmvVowels("aBBa.123")
>>>'BB.123'
>>>rmvVowels(123.23)
>>>''
```

[7]

```
#this was done in class
 #other solutions possible, of course
   #e.g.  using str.replace() - 5 times
   if type(worda)!=str: #[1]
      return "" #[1]
   wordb='' #[1]
   for i in worda:  #[1]
      if i in 'aeiou': #[1]
         continue
      wordb+=i #[1]
```

return wordb #[1]

# function to be rewritten

(e) The string function `str.upper()` (used in `Q2.py`) converts alphabetic characters from upper to lowercase (i.e. small letters to capitals). You could write your own `upper()` function using the builtin Python `ord()` and `chr()` functions. Explain how you would do this (You do not need code, just a clear explanation and/or the algorithm). [5]

The function ord(0 returns the Unicode corresponding to a character.[1] Alphabetic characters are stored sequentially in Unicode [1], with capitals appearing before lowercase. To convert from lower to upper, you need to step through the string character by character[1]. For each character in the Unicode range for lowercase, you need to substract the unicode difference between 'a' and 'A': chr(ord(char)-(ord('a')-ord('A')))[2]

Question 3 [10]

For each question below, **write down ONE letter** corresponding to the correct answer.

(a) Which of these can be a black box technique?

a. path testing

b. equivalence classes

c. statement coverage

d. debugging

e. tracing

B

(b) An empty list

a. `[]`

b. `0`

c. `[' ']`

d. `[0]`

e. none of the above

A

(c) If `X=[[[1,2],[3,4]],[5,6]]`, what is `X[0][1][1]`?

a. 2

b. 3

c. 4

d. 5

e. 6

C

(d) What element is typically NOT found in simple recursive functions?

a. base case

b. recursive step

c. loop

d. function definition

e. parameter

C

(e) What data types can be used with recursion?

a. strings

b. integers

c. floats

d. all of the above

e. none of the above

D

(f) What is NOT an element of an exception mechanism used with files?

a. finally

b. try

c. IOError

d. except

e. else

E

(g) What is the time complexity (in comparisons) of the binary search algorithm?

a. O (1)

b. O (log n)

c. O (n)

d. O (n log n)

e. O (n^2)

B

(h) What is the time complexity (in comparisons) of the worst case quick sort algorithm?

a. O (1)

b. O (log n)

c. O (n)

d. O (n log n)

e. O (n^2)

E

(i) What is the floating point value of the following IEEE 754 single recision number?
0 10000001 11001000000000000000000

a. 3.0625

b. -3.0625

c. 7.125

d. 7.5

e. -7.25

C

(j) What function does this truth table represent?

A	B	F
0	0	0
0	1	0
1	0	1
1	1	0

a. A OR B

b. A AND B

c. NOT (A OR B)

d. NOT (A AND B)

e. A AND (NOT B)

E

Question 4 [20]

Examine the following program and answer the questions that follow.

```
def reverse (l):
    new_list = []
    for item in l:
        new_list.insert (0, item)
    return new_list

test_list = [90, 66, 79, 92, 97, 80, 29, 83, 51, 24]
rev_list = reverse (test_list)
```

(a) Suppose that we want to test the **reverse** function by itself. What are the equivalence classes and boundary values? [2]

equivalence class: all non-empty lists [1]

boundary: empty list [1]

(b) If statement coverage is used as a testing technique, what is the minimum number of test cases needed to test the **reverse** function? [1]

1 [1]

(c) Rewrite the **reverse** function using recursion. [4]

```
def reverse2 (l):
    if len (l) == 0:[1]
        return [] [1]
    else:
        return reverse2 (l[1:]) + [l[0]] [2]
```

(d) Write Python code to write the contents of **test_list** to the text file named `output.txt`. [4]

```
f = open ("output.txt", "w") [1]
for item in test_list: [1]
    print (item, file=f) [1]
f.close () [1]
```

(e) Show the steps of the merge sort algorithm when applied to sort **test_list**. [3]

[90, 66, 79, 92, 97][80, 29, 83, 51, 24]

[90, 66, 79][92, 97][80, 29, 83][51, 24]

[90, 66][79][92][97][80, 29][83][51][24]

[90][66][79][92][97][80][29][83][51][24]

[66, 90][79][92][97][29, 80][83][51][24]

[66, 79, 90][92, 97][29, 80, 83][24, 51] [1]

[66, 79, 90, 92, 97][24, 29, 51, 80, 83] [1]

[24, 29, 51, 66, 79, 80, 83, 90, 92, 97] [1]

(f) Convert the decimal number 90 to binary using repeated division. Show all calculation. [3]

90 / 2 = 45 r 0

45 / 2 = 22 r 1

22 / 2 = 11 r 0

11 / 2 = 5 r 1

5 / 2 = 2 r 1

2 / 2 = 1 r 0

1 / 2 = 0 r 1

Answer = 1011010

(g) Calculate 7-12 (both decimal) using 8-bit 2's complement binary addition. Show all calculation. [3]

7 – 12

= 00000111 + 2's complement (00001100) [1]

= 00000111 + 11110100 [1]

= 11111011 [1]

= - 5

Question 5 [10]

During Fresher's Week 2015, the UCT Mountain and Ski club wants to hold a “treasure hunt” competition every day to raise funds. Each day, the “treasure” is hidden at a new random point on a map. Contestants will pay to choose a point on the map, by specifying a latitude and longitude. At the end of each day, the contestant whose point is closest to the treasure will win. The Mountain and Ski club have chosen you to write the software to run this contest.

Your program must ask for a sequence of names and guessed coordinates. After the sequence has been completed, your program must output the winning name and coordinates.

Complete the Python code below to do this. The upper and lower bounds for latitudes and longitudes are provided as parameters.

Remember that the distance between 2 points - (x1, y1) and (x2, y2) - is defined as follows:

$$d=\sqrt{(x2-x1)^2+(y2-y1)^2}$$

```
import random
import math
def contest(latt_min,latt_max,long_min, long_max):
    prize_x=random.randint(latt_min,latt_max)
    prize_y=random.randint(long_min,long_max)
```

```
import random
import math

def contest(latt_min,latt_max,long_min, long_max):
  prize_x=random.randint(latt_min,latt_max)
  prize_y=random.randint(long_min,long_max)
  name=input("Enter student number for next contestant:")
  Winner=name
  dist=math.sqrt((latt_max-latt_min)**2 + (long_max-long_min)**2)
  while name:
    x=eval(input("Type in lattitude:”))
    y=eval(input("Type in longitude:”))
    new_dist=math.sqrt((x-prize_x)**2 + (y-prize_y)**2)
    if new_dist<dist:
      dist=new_dist
      Winner=name
    name=input("Enter student number for next contestant:")
  print("The winner is:",Winner)
  print("Prize at:",prize_x,prize_y)
```

contest(0,100,0,100)

distance calculation [2]
finding closest point [2]
tracking name [2]
input [2]
loops and general structure [2]

Code examples – you may detach this sheet.

Question 1

```
#Module Q1.py
def chop(item,pos):
     if type(item)==int:
          x=item//pos
          item=x
     else:
          item=item[0:pos]
     print(item)

def swell(item):
     if type(item)==int or type(item)==float:
          return item*2.0
     elif item.isdigit():
          item=eval(item)*2
     return item*2
```

Question 2

```
#Module Q2.py
def ana(a,b):
    a,b=str(a),str(b)
    if len(a)==len(b):
        a,b=a.upper(),b.upper()
        while a and b:
            if b[0] not in a:
                break
            delete_pos = a.find(b[0])
            a=a[:delete_pos]+a[delete_pos+1:]
            b=b[1:]
        else: return True
    return False

def rmvVowels(worda):
    pass
    # function to be rewritten
```