Name:

____

____

Student Number:

____

Please fill in your Student Number and Name.

Student Number : ____________________________________

University of Cape Town ~ Department of Computer Science

Computer Science 1015F ~ 2018

Examination

Question	Max	Internal	External
1	**20**		
2	**10**		
3	**10**		
4	**10**		
5	**10**		
6	**10**		
TOTAL	**70**		

Marks : 70

Time : 120 minutes

Instructions:

a) Answer all questions.

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

c) You may use a calculator – BUT show all calculations where

Question 1 – Multiple Choice [20 Marks]

1. What is an example of an identifier?
   a. 3
   b. #name
   c. _biscuit
   d. Number
   e. c and d

Answer: e

2. What is the exact output of the below statements?

```
a = 6
a = a / 2
print(a)
```

   a. a
   b. “a”
   c. 3
   d. 3.0
   e. None of the above

Answer: d

3. The CPU follows which process?
   a. The instruction cycle
   b. The process paradigm
   c. The execute paradigm
   d. The fetch-execute cycle
   e. The compute cycle

Answer: d

4. What is described as the ‘brain of the computer’?
   a. CPU
   b. RAM
   c. HDD
   d. GPU
   e. Motherboard

Answer: a

5. What is an example of a relational operator?
   a. //
   b. <
   c. not
   d. !=
   e. B and D

Answer: e

6. Suppose

```
def nPrint(message, n):
   while (n > 0):
     print(message,end=’ ’)
     n--;
```

What is the output of the call `nPrint('a', 4)`?
   a. aaaaa
   b. aaaa
   c. aaa
   d. a a a a
   e. a a a

Answer: d

7. Analyze the following code:

```
def main():
     print(xMethod(5, "500L"))


def xMethod(n, l):
     print("int, long");
     return n;

main()
```

   a. The program displays int, long followed by 5 on a new line.
   b. The program runs fine but displays things other than given in a.
   c. The program does not compile because the compiler cannot invoke xMethod.
   d. The program displays int, long followed by 5 on the same line.
   e. The program displays 5, 500L followed by 5 on a new line.

Answer: a

8. A variable defined inside a function is referred to as __________.
   a. a global variable
   b. a function variable
   c. a constant variable
   d. a local variable
   e. a block variable

Answer: d

9. What would be the result of attempting to compile and run the following code?

```
x = [1, 2, 3]
print("Value is:", x[1])
```

a. The program has a syntax error because the syntax [1, 2, 3] is wrong and it should be replaced by {1, 2, 3}.
b. The program has a syntax error because the [1, 2, 3] is wrong and it should be replaced by [0,1, 2, 3].
c. The program has a syntax error because the [1, 2, 3] is wrong and it should be replaced by [1.0, 2.0, 3.0].
d. The program compiles and runs fine and the output "Value is 1" is printed.
e. The program compiles and runs fine and the output "Value is 2" is printed.

Answer: e

10. Consider the following Python program.

```
def Cat(x):
  if (x<0): return 0
  if (0<x<100):return x
  if (x>100): return 100
```

Select the set of inputs below that will comprise a **complete**, but **minimal**, **path coverage test** of this function.

a. -200
b. -20, 65, 100
c. -40, 0, 78, 200
d. -50, 0, 65, 100, 200
e. -80, 0, 45, 67, 100, 200

*Answer:**c***

11. Consider the following Python program.

```
def SumList(lst):
  for a in lst:
    total=total+a
  return total
```

a. The function has a **syntax error.**
b. The function has a **runtime error.**
c. The function has a **logic error.**
d. All of the above.
e. None of the above.

*Answer:**b***

12. What output does the Python code below produce?

```
X=[]
for i in range(10):
    X.append(i)
print(X[9])
```

a. 9
b. 10
c. A runtime error
d. [1, 2, 3, 4, 5, 6, 7, 8, 9, 10]
e. X[9]

*Answer:**a***

13. What output does the Python code below produce?

```
X,Y=[],[1,2,3]
X.append(Y)
X.append(Y)
print(X[1][2])
```

a. []
b. [1, 2, 3]
c. 3
d. 2
e. A runtime error

*Answer:**c***

14. What output does the Python code below produce?

```
def Recur(n):
  if n<=0: return '0'
  text=Recur(n-1)
  return str(n)+text+str(n)
print(Recur(5))
```

a. RecursionError
b. 00000
c. 012345
d. 543210
e. 54321012345
f. 43211234

Answer: **e**

15. Which of the functions below calculates the nth Fibonacci number?

a.

```
def Eg1(n):
  if n==1 or n==2: return 1
  return Eg1(n-1)+Eg1(n-2)
```

b.

```
def Eg2(n):
    if n==0: return 1
    return n*n+Eg2(n-1)
```

c.

```
def Eg3(n):
    if n==0: return 1
    return n*Eg3(n-1)
```

d.

```
def Eg4(n):
    if n==0: return 0
    return n+Eg4(n-1)
```

e.

```
def Eg5(n):
    if n==0: return 0
    return Eg5(n-1)+Eg5(n-2)
```

Answer: a

16. What is the **time complexity** of the **merge sort** algorithm in the **average case**?
    a. O(1)
    b. O(n)
    c. O(n+log n)
    d. O(n log n)
    e. $O(n^2)$

Answer:d

17. Which of the following is a **stable** sorting algorithm**?**

    a. Selection sort
    b. Merge sort
    c. Quick sort
    d. Bubble sort
    e. Binary search

---

*Answer: **b***

18. The number –

    $1111\ 1111_2$

    - in 8-bit **one's complement binary** representation is equivalent to:

    a. 128_{10}
    b. 127_{10}
    c. -127_{10}
    d. -1_{10}
    e. 0_{10}

---

*Answer:**e***

19. What is the decimal floating point value of the following IEEE 754 single precision number?
    1 10000000 10000000000000000000000

    a. 10
    b. 129
    c. -0.5
    d. -1
    e. -3

---

*Answer:**e***

20.

A	**B**	**C**	***F***
0	0	0	0
0	0	1	1
0	1	0	0
0	1	1	1
1	0	0	0
1	0	1	1
1	1	0	1
1	1	1	1

For this truth table, what is the **Boolean expression** represented by *F*?

a. C
b. A AND B OR C
c. A AND B AND C
d. A AND B OR B AND C
e. A OR B OR C

---

*Answer:**b***

Question 2 [10 Marks]

Lucy has written the following code.

```
bored = input(“Are you  bored right now?”)
game = input(“Do you want to play a game?”)
if bored == “Yes” and game == “No”:
    print(“Why no games? You said you were bored :( “)
if bored == “Yes” or “Y”:
     print(“I’m sorry you’re bored.”)
else:
     print(“Okay”)
```

a) Write down the exact output that would displayed if `bored` = “Yes” and `game` = “No”. (2)

Answer: “Why no games? You said you were bored :(“ [1]
“I’m sorry you’re bored” [1]

b) Give values for `bored` and `game` that would lead to the else statement being executed. (2)

Answer: There are none [1], because the “or” is always True [1].

c) Give an alternative to Lucy’s first `if` statement. (2)

Answer: if bored == “Yes”:
if game==”Yes” or game == “Y”:
print(“Why no games? You said you were bored :(“)
[1] for 2 if statements, [1] for correct indentation

d) What is the difference between primary and secondary storage? Give an example of each. (4)

Answer: Primary storage: data not lost when PC switched off [1]. E.g. HDD/SSD [1]. Secondary storage: data lost when PC switched off [1]. E.g. RAM [1].

Question 3 **[10 Marks]**

a) What is the output of the following code? (3)

```
  def swap1(a,b):
    temp = a
    a = b
    b = temp

  def swap2(a):
    temp = a[0]
    a[0] = a[1]
    a[1] = temp

  a = [1, 2]
  swap1(a[0],a[1])
  print("a[0] = " + str(a[0]) + " a[1] = " + str(a[1]))
  swap2(a)
print("a[0] = " + str(a[0]) + " a[1] = " + str(a[1]))
```

Answer:
a[0] = 1 a[1] = 2
a[0] = 2 a[1] = 1 ***[1 mark for correct string format, 1 mark each for correct values for a[0] and a[1] on each line]***

b) Write a function that take in a string and returns the string with every third letter (assuming all are letters) in uppercase. (7)

```
def up(a): [1 mark for correct header]
  b='' [1 mark]
  for i in range(len(a)): [1 mark for correct loop]
    if (i+1)%3==0: [1 mark for every third element]
    b+=a[i].upper() [1 for string concatenation, 1 for .upper()]
  else:
    b+=a[i]
return b [1 for returning correct value]
```

Question 4 **[10 Marks]**

Examine the `q4.py` module listed at the end of this examination and answer the following questions.

a) What would be the values of `person` and `name_list` if you executed the code: `person = name_list.pop(1)`? [2]

person would be equal to: ["Daenerys","Targaryen"]

name_list would be equal to:
[["Jon","Snow"],["Arya","Stark"],["Tyrion","Lannister"],["Sansa","Stark"], ["Bran","Stark"], ["Cersei","Lannister"]]

b) Write down the exact output that is printed when this module is executed. [2]

[["Arya", "Stark"],["Sansa", "Stark"],["Bran", "Stark"]]

c) Write down a **minimal** set of inputs to the `mystery()` function that would constitute a complete **path test** of the function. [2]

Two sets of inputs required:

- *a 2d list and a value that appears at least once at index 1 for an inner list within the outer list*
  eg. [["John", "Smith"],["Tom", "Jones"]], "Smith"

- *a 2d list and a value that does not appear at index 1 for any of the inner lists within the outer list*
  eg. [["John", "Smith"],["Tom", "Jones"]], "Ndlozi"

d) Write down the code needed to write the items in `name_list` to a text file `names.txt` in such a format that the text in the file looks like this:

```
Snow, Jon
Targaryen, Daenerys
Stark, Arya
Lannister, Tyrion
Stark, Sansa
Stark, Bran
Lannister, Cersei
```

[4]

```
f = open("names.txt", "w")
for item in name_list:
    print(item[1]+", "+item[0], file = f)
f.close()
```

Question 5 **[10 Marks]**

Examine the **`Q5.py` module** listed at the end of the exam paper and answer the following questions.

a) Write down the exact output to the screen when this module is executed, assuming that the `quick_sort` function operates correctly.

[3]

Answer:
[3, 6, 9, 10, 0]
[0, 3, 6, 9, 10]
[0, 3, 6, 9, 10]
#1 mark each – key is recognising that the array alters the original

b) The function `quick_sort` has some lines hidden. Write down the hidden lines here, so that the function will operate correctly. [4]

Answer:

```
if stop > start: [1]
    pivot = partition (values, start, stop) [1]
    quick_sort2 (values, start, pivot-1) [1]
    quick_sort2 (values, pivot+1, stop) [1]
```

c) You are applying for a job and, as part of the interview process, your prospective employer asks you to write a function to sort very large files. They suggest using the `other_sort` function in the Q5.py module instead of `quick_sort`. Is this a good idea? Justify/explain your answer. [3]

Answer: No. The other sort is a mergesort and not in place – will use double the space, which is not good for very large arrays.

Question 6 [10 Marks]

You are working in a holiday job for an academic publisher. They have asked you to correct manuscripts to conform to a particular style. As a first task, they want you to ensure that **all numbers less than ten** are written as the word and not the digits in all manuscripts. For example 'nine', not '9'; 'eight', not '8' and so on. You are finding the search-and-replace task very tedious, and so want to write a Python function to automate this. As all the manuscripts are plain text files, your function `correct_text` will read in the text from a file (whose name provided to the function) and write out the text with all the digits corrected to another file (whose name is also provided to the function). The `correct_text` function should print a helpful message if the input file does not exist.

For example, if the function is called as follows:

```
correct_text(“input.txt”,”output.txt”)
```

and the file “input.txt” contains the following text:

```
On the fifth day of Christmas
my true love sent to me:
5 Golden Rings
4 Calling Birds
3 French Hens
2 Turtle Doves
and 1 Partridge in a Pear Tree
```

the output file will then be written as:

```
On the fifth day of Christmas
my true love sent to me:
five Golden Rings
four Calling Birds
three French Hens
two Turtle Doves
and one Partridge in a Pear Tree
```

However, if the file “input.txt” does not exist, this message will be displayed instead:

```
File input.txt does not exist!
```

Now complete the `correct_text` function on the following page.

```
def correct_text(input_file_name,output_file_name):
```

Answer: #there are correct alternatives

```
def correct_text(input_file_name,output_file_name):

    swaps={'1':"one",'2':"two",'3':"three",
'4':"four",'5':"five",'6':"six",'7':"seven",'8':"eight",'9':"nine"} #[1] for dictionary, but can do another wat

    try:
        #open file for reading
        inp=open(input_file_name,'r') #[1]
        allText=inp.read() #[1]
        inp.close() #[0.5]

        #4 marks for some correct way to replace the words
        for word in swaps: #[1]
            textList=allText.split(word) #[1]
            allText=swaps[word].join(textList) #[1]

        # writing out to file correctly
        out=open(output_file_name,'w') #[1]
        print(allText,file=out) #[0.5]
        out.close() #[0.5]
    except IOError: #error handling [1.5]
        print("File",input_file_name," does not exist")

correct_text("input.txt","output.txt")
```

Code examples for the examination (you may detach these sheets).

Code for Question 4

```
  #q4.py

def mystery(arr, s):
    temp = []
    for item in arr:
        if s == item[1]:
            temp.append(item)
    return temp

name_list = [["Jon","Snow"],["Daenerys","Targaryen"],
["Arya","Stark"],["Tyrion","Lannister"],
["Sansa","Stark"],["Bran","Stark"], ["Cersei","Lannister"]]

new_list = mystery(name_list, "Stark")
print(new_list)
```

Code for Question 5

```
def swap ( values, source, dest ):
   values[source], values[dest] = values[dest], values[source]

def partition ( values, start, stop ):
   pivot = values[stop]
   midpoint = start
   for position in range (start, stop):
      if values[position] <= pivot:
         swap (values, position, midpoint)
         midpoint += 1
   swap (values, midpoint, stop)
   return midpoint

def quick_sort ( values, start, stop ):
   """Sort values using quicksort algorithm."""
   #Code hidden here

def combine ( list1, list2 ):
   new_list = []
   while len(list1)>0 and len(list2)>0:
      if list1[0] < list2[0]:
         new_list.append (list1[0])
         del list1[0]
      else:
         new_list.append (list2[0])
         del list2[0]
   return new_list + list1 + list2

def other_sort ( values ):
   if len(values)>1:
      sorted1 = other_sort (values[:len(values)//2])
      sorted2 = other_sort (values[len(values)//2:])
      return combine (sorted1, sorted2)
   else: return values

X=[10,9,3,6,0]
quick_sort(X,0,3)
print(X)
quick_sort(X,0,len(X)-1)
print(X)
quick_sort(X,0,3)
print(X)
```