Office Use Only

MONASH University

Semester Two 2017
Examination Period
Faculty of Information Technology

EXAM CODES: **FIT1045**

TITLE OF PAPER: **ALGORITHMS AND PROGRAMMING FUNDAMENTALS IN PYTHON**

EXAM DURATION: 2 hours writing time

READING TIME: 10 minutes

THIS PAPER IS FOR STUDENTS STUDYING AT: (tick where applicable)

□ Berwick ✓Clayton ✓ Malaysia □ Off Campus Learning □ Open Learning
□ Caulfield □ Gippsland □ Peninsula □ Monash Extension □ Sth Africa
□ Parkville □ Other (specify)

During an exam, you must not have in your possession any item/material that has not been authorised for your exam. This includes books, notes, paper, electronic device/s, mobile phone, smart watch/device, calculator, pencil case, or writing on any part of your body. Any authorised items are listed below. Items/materials on your desk, chair, in your clothing or otherwise on your person will be deemed to be in your possession.

No examination materials are to be removed from the room. This includes retaining, copying, memorising or noting down content of exam material for personal use or to share with any other person by any means following your exam.
Failure to comply with the above instructions, or attempting to cheat or cheating in an exam is a discipline offence under Part 7 of the Monash University (Council) Regulations.

AUTHORISED MATERIALS

OPEN BOOK □ **YES** ✓ **NO**

CALCULATORS □ **YES** ✓ **NO**

SPECIFICALLY PERMITTED ITEMS □ **YES** ✓ **NO**
if yes, items permitted are:

Candidates must complete this section if required to write answers within this paper

STUDENT ID: __ __ __ __ __ __ __ __ DESK NUMBER: __ __ __ __ __

This page intentionally left blank, use if needed but it will not be marked.

Important Information

Any **algorithms** you produce should be **written in words**, you will be told to write a python program or function if you are to write in python.

All Python code you write for this exam must satisfy the following requirements:

- Syntax should satisfy Python 3 requirements
- Use syntax, functions, structures and constructs presented in lectures.
- avoid using any inbuilt python functions/methods which make any tasks significantly simpler

Write down any assumptions you make.

Do not write anything in this table. It is for office use only.

Question	Points	Score
1	4	
2	4	
3	6	
4	5	
5	6	
6	6	
7	6	
8	6	
9	9	
10	10	
11	8	
Total:	70	

This page intentionally left blank, use if needed but it will not be marked.

Circle one letter for each part corresponding to the correct response

If you change your mind, clearly cross out your previous choice and circle the new one and write the letter chosen on the side

Question 1: [4 marks]

(a) (1 mark) what will be printed by the following code?

```
initial = 123
pos = 0
s=0
while pos < len(str(initial)):
        v = str(initial)[pos]
        s = s + int(v)
        pos = pos + 1
print(s)
```

A. 6

B. 3

C. 123

D. 0

E. nothing; an error would occur or the code would not terminate

(b) (1 mark) what will be printed by the following code?

```
aList = ["1","2"]
bList = ["3","2"]
cList = aList + bList
value = ""
for index in range(len(cList)):
        value = cList[index] * index
print(value)
```

A. ""

B. "222"

C. "33"

D. "233222"

E. nothing; an error would occur or the code would not terminate

2

This page intentionally left blank, use if needed but it will not be marked.

(c) (1 mark) what will be printed by the following code?

```
aString = "my␣string␣is␣too␣short"
while len(aString) > 10:
        aString = aString + "!"
print(len(aString))
```

A. 0

B. 10

C. 11

D. 22

E. nothing; an error would occur or the code would not terminate

(d) (1 mark) What will be printed after the following code is run?

```
def whatIsThis(aList,p1,p2):
        tmp = aList[p1]
        aList[p1] = aList[p2]
        aList[p2] = tmp


myList = [10,20,30,40]
whatIsThis(myList,1,3)
print(myList)
```

A. [10,20,30,40]

B. [10,20,30,20]

C. [30,20,10,40]

D. [10,40,30,20]

E. nothing; an error would occur or the code would not terminate

2

This page intentionally left blank, use if needed but it will not be marked.

Question 2: [4 marks]

For each of the following, list the complexity and explain why it has this complexity.

(a) (2 marks) assume N is len(bList)

```
def bFunc(bList):
        b = 1
        k = 0
        while k < len(bList):
                b = b * bList[k]
                k = k + 2
        return b
```

(b) (2 marks) assume N is len(cList)

```
def cFunc(cList):
        c = 1
        k = 0
        half = len(cList)//2
        while k < len(cList):
                c = c - cList[k]
                k = k + half
        return c
```

This page intentionally left blank, use if needed but it will not be marked.

Question 3: [6 marks]

This question is about stacks and queues.

(a) (3 marks) Consider a stack as below:

```
bottom -> [5, 20, 3] <- top
```

Assuming that the right end of the above list represents the top of the stack, draw the state of the stack after each of the following operations are run on it (0.5 marks per correct state).

Be sure to make clear what the top of the stack is in each diagram.

1. push 5
2. push 10
3. pop
4. pop
5. pop
6. push -5

3

This page intentionally left blank, use if needed but it will not be marked.

(b) (3 marks) For the following program, show the state of the queue Q after each iteration of the loop. You may assume the following:

- append(aQueue,A) - will append A into the queue aQueue
- serve(aQueue) - will serve from the queue aQueue
- size(aQueue) - will return the number of items in the queue aQueue
- the left end of Q is the front and the right end of Q is rear

```
def maxInQueue(aQueue):
        N = size(aQueue)
        theMax = 0
        while N > 0:
                item = serve(aQueue)
                if item > theMax:
                        theMax = 0.5*item
                        append(aQueue,theMax)
                N -= 1
        return theMax

Q = [1,15,6,8,3,1]
print(maxInQueue(Q))
```

3

This page intentionally left blank, use if needed but it will not be marked.

Question 4: [5 marks]

This question is about sorting and algorithms.

(a) (1 mark) Given a list $L = [1, 6, 1, 6, 9, 2, 4, 2, 6, 1, 2, 1]$ show what list counting sort would create in order to sort this list. Be sure to specify what the indices and values represent.

(b) (4 marks) Give an **algorithm** which describes how selection sort sorts a list of numbers

This page intentionally left blank, use if needed but it will not be marked.

Question 5: [6 marks]

Select an algorithm (such as the algorithm you wrote for selection sort earlier) and **explain** why it is an algorithm. Be sure to consider each of the properties of algorithms (that we discussed in lectures) in your answer.

6

This page intentionally left blank, use if needed but it will not be marked.

Question 6: [6 marks]

Consider the knapsack problem for a maximum weight of 6kg for the following set of items.

Table 1: items available for knapsack

Item	Spoon	Candlestick	Urn	Television
Weight	1	2	4	5
Value	80	150	900	1000

Assuming this problem was solved using the backtracking method, show a back-tracking diagram for this problem. This should clearly show the partial solutions, next options and complete solutions.

For convenience, you may shorten the names of spoon, candlestick, urn and television to s, c, u and t respectively.

This page intentionally left blank, use if needed but it will not be marked.

Question 7: [6 marks]

This question is about the transform and conquer strategy.

Given the maxheap represented as an array as [7,4,1,0,3,1]. Show what happens to the heap by performing the extract maximum method **twice**. Ensure you show the state of the heap (in either array or tree representation) after every swap. If at any point no swap occurs, explain why.
Reminder: the children of any node in the array representation can be found at $2 \times i$ *and* $1 + 2 \times i$ *(where indexing begins at* 1*)*

6

This page intentionally left blank, use if needed but it will not be marked.

Question 8: [6 marks]

Give a useful invariant for the inner loop in terms of j and k and the outer loop in terms of k and list what this code does. [2+2+2=6]

```
def cFunc(cList):
    k=1
    while k < len(cList):
        j = k-1
        while j >= 0:
            if cList[j] > cList[j+1]:
                tmp = cList[j]
                cList[j] = cList[j+1]
                cList[j+1] = tmp
            j = j - 1
        k = k + 1
    return cList
```

6

This page intentionally left blank, use if needed but it will not be marked.

Question 9: [9 marks]

This question is about recursion and divide and conquer approaches to problem solving. Consider the task of counting instances of a particular character in a string S.

(a) (6 marks) Consider the recursive definition given below:

$$Count(i, S, target) = Count(i-1, S, target) + \begin{cases} 1 \text{ where } S[i] \text{ is target} \\ 0 \text{ where } S[i] \text{ is not target} \end{cases}$$

Note that $Count(i, S, target) = 0$ *where* $i < 0$
Eg. consider the string $S =$`"ab cbb ef b gh ijkl"` with a target of `"b"`. Here:
$count(0, S, target) = 0$ as position 0 is not b
$count(1, S, target) = 1$ as position 1 is b
$count(2, S, target) = 1$ as position 2 is not b
$count(3, S, target) = 1$ as position 3 is not b
$count(4, S, target) = 2$ as position 4 is b
$count(5, S, target) = 3$ as position 5 is b
write a **recursive python function** *Count* which accepts as input the position, i, string, S, and target, *target*, and implements this relationship, returning the result. *Note: you do not need to consider the case of running count with an i value exceeding the number of characters in the string*

6

This page intentionally left blank, use if needed but it will not be marked.

(b) (3 marks) Describe in words how the divide and conquer approach can be applied to this problem.

3

This page intentionally left blank, use if needed but it will not be marked.

Question 10: [10 marks]

Consider the problem of finding the smallest vertex cover for a given graph G. This question has you considering a greedy approach and contrasting with a brute force solution.

(a) (7 marks) Write an **algorithm** ($greedyCover$) which accepts a graph (G) and, using the principle of greed, finds a set of vertices corresponding to the smallest vertex cover (or something close to it). Your algorithm should output the vertices in the vertex cover found.

Note: Ensure you clarify where your algorithm is applying greed. If you are unsure how to write this as an algorithm, you can still get up to 3 marks by explaining how you were intending to apply greed in this situation

*Note2: Given a greedy approach will not always give an optimal solution to this problem, your algorithm is not expected to consistently give the **best** answer but it should still provide a vertex cover and demonstrate understanding of greed.*

This page intentionally left blank, use if needed but it will not be marked.

(b) (3 marks) Explain in words what a brute force solution to this problem would entail.

This page intentionally left blank, use if needed but it will not be marked.

Question 11: [8 marks]

Consider a variant of the N-Queens problem which aims to find a solution to N-Queens which also satisfies the restriction that if you add together the row positions of the queens in the first two columns on the board, they add to some value k. For example, the 6 queens problem and a k value of 7, there is just one solution (of the four solutions to 6 Queens) which meets this restriction.

(a) (1 mark) Explain what a certificate to this problem must include.

(b) (5 marks) Explain in words how you would check the certificate for this problem for a k value of 10.

This page intentionally left blank, use if needed but it will not be marked.

(c) (2 marks) Using **only** your answer to the previous part, argue as to whether this variant of the N-Queens problem is a P or NP class of problem. Justify your answer.

2