| Mansoura University | | Faculty of Engineering | | Biomedical Engineering Dept. | |
|---|---|---|---|---|---|

**Course Title: Algorithms and Data Structure** **Course Code: CSE 152** **Total Marks: 50 Marks**
**Date: Aug. 29, 2018 (Final Exam)** **Allowed time: 2 hrs** **No. of Pages: (2)**

Welcome to the final Exam for ***Algorithms and Data Structure***. Read each problem carefully. There are five required problems (ordered as course syllabus sequence - each worth 10 points- Extra 10 points). Please use a separate sheet for the answer to each question. Assume any missing data. Good luck and be sure to show your work!

## QUESTION NO. (1) (10 MARKS) [INTRODUCING DATA STRUCTURES AND ALGORITHMS]

Q1-A) What is a Data Structure? What are the advantages and disadvantages of the various data-structures? 5

Q1-B) What is an algorithm? Give some practical examples of using Algorithms. List the major steps for developing an algorithm. 3

Q1-C) What are the criteria of algorithm analysis? What is meant by time and space complexity of algorithm? 2

Q1-D) What is meant by Data abstraction and ADT? What data structure implements LIFO behaviour? 2

## QUESTION NO. (2) (10 MARKS) [STRING-LISTS-TUPLES-DICTIONARY]

Q2-A) List the different operations (at least three) which can be performed on (String-Lists-tuples-Dictionary). 3

Q2-B) Write a Python program to multiplies all the items in a list. 2

Q2-C) Write a Python program to find maximum and the minimum value in a set. 2

Q2-D) Write a Python program to sort (ascending and descending) a dictionary by value. 3

## QUESTION NO. (3) (10 MARKS) [ARRAYS]

Q3-A) What are the Similarities and differences between the array and the list? 2

Q3-B) Define one-dimensional array and two-dimensional array. 2

Q3-C) What are Modules? Why we use Modules in python? What is the ctypes and NumPy Module. 3

Q3-D) Write a Python program to reverse the order of the items in the array. 3

## QUESTION NO. (4) (10 MARKS) [ALGORITHM ANALYSIS]

Q4-A) What is the difference between Algorithms – Code – Pseudocode? 3

**Q4-B) Evaluate the complexity following Python Code:**

```
def ex1( n ):
  total = 0
  for i in range( n ) :
    total += i
  return total
```

```
def ex3( n ):
  count = 0
  for i in range( n ) :
    for j in range( n ) :
      count += 1
  return count
```

3

**Q4-C) For each function *f(n)* below, give an asymptotic upper bound using "big-Oh" notation. You should give the tightest bound possible.**

4

a) $f(n) = 100\,n^3 - 7n^3 + 14n^2$
b) $f(n) = 100\,n^3 - 100n^3 + 7n^2$
c) $f(n) = \log(7\,n^2)$

## QUESTION NO. (5) (20 MARKS) (DYNAMIC DATA STRUCTURES)

**Q5-A)** Write the Pseudocode **Add (value)** for the Singly Linked List. 4

**Q5-B)** We can implement each of the stack operations with just a few lines of code. Write the Pseudocode for *STACK-EMPTY(S), PUSH(S,x), Pop(S,x).* 4

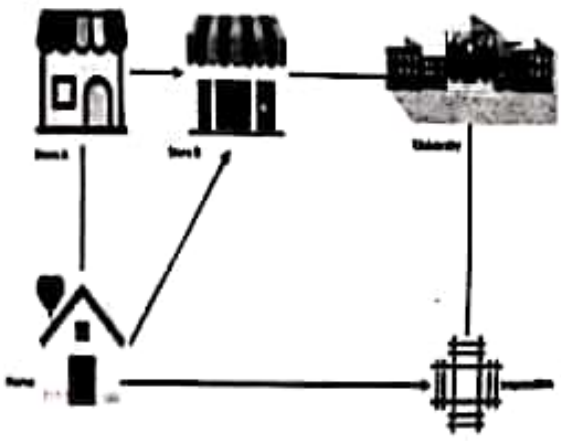
**Q5-C)** Suppose q is an instance of the Queue class and assume that the array implementation is used. Also, assume the size of the array is 3. Show q after all of the following operations have been completed. Assume the queue is empty to start with. Show how the front, rear, and elements change. 4
**(i)** q.enqueue(39); **(ii)** q.enqueue (22) ; (iii) item1= q. dequeue( ); **(iv)** q.enqueue(59); **(v)** item2= q. dequeue( ); **(vi)** item3= q. dequeue( )

**Q5-D)** Show the stack elements after the following operations have all been completed (Draw the final picture of **s.elements[ ]** and show the value of **s.top**). 4
**(i)** s.push (20); (ii) s.push (51); (iii) s.pop(); **(iv)** s.push (43).

**Q5-E)** What is the difference between Queue and Priority Queue? 2

**Q5-F) For the following figure:** 2

- List two data structures to implement a map.
- Write the steps to find the shortest path from home to the university.
- What is the relation between data structure and the algorithm used to find the shortest path form home to the university?





*Best of luck*

*Dr. M.Badawy*

Course Title: Algorithms and Data Structure | Course Code: CSE 152 | Total Marks: 50 Marks
Date: Aug. 29, 2018 (Final Exam) | Allowed time: 2 hrs | No. of Pages: (2)

## Model Answer

**QUESTION NO. (1) (10 MARKS) [INTRODUCING DATA STRUCTURES AND ALGORITHMS]**

**Q1-A) What is a Data Structure? What are the advantages and disadvantages of the various data-structures?**

- A data structure is an arrangement of data in a computer's memory (or sometimes on a disk).
- A group of data elements grouped together under one name. These data elements, known as members, can have different types and different lengths.

| *Data Structure* | *Advantages* | *Disadvantages* |
|---|---|---|
| Array | Quick insertion, very fast access if index known. | Slow search, slow deletion, fixed size. |
| Ordered array | Quicker search than unsorted array. | Slow insertion and deletion, fixed size. |
| Stack | Provides last-in, first-out access. | Slow access to other items. |
| Queue | Provides first-in, first-out access. | Slow access to other items. |

**Q1-B) What is an algorithm? Give some practical examples of using Algorithms. List the major steps for developing an algorithm.**

- **Algorithm:** Step-by-step procedure for performing some task in a finite amount of time.
- Some practical examples of using Algorithms: **Search Engines - GPS navigation Self-Driving Cars - E-commerce - Banking - Medical diagnosis- Robotics**
- **The major steps for developing an algorithm.**
  - **Design the algorithm:** correctness and efficiency
  - **Verify and prove correctness** :will it work for all cases?
  - **Analyze the algorithm:** how long will it run?
  - **Implement the algorithm:** coding and programming

**Q1-C) What are the criteria of algorithm analysis? What is meant by time and space complexity of algorithm?**

- An algorithm is said to be efficient and fast, if it takes less time to execute and consumes less memory space.
- The performance of an algorithm is measured on the basis of following properties :

1. Time Complexity: how long will it run?
- 2. Space Complexity: how memory and CPU will it use?

**Q1-D) What is meant by Data abstraction and ADT? What data structure implements LIFO behaviour?**

- **An abstraction** is a mechanism for separating the properties of an object and restricting the focus to those relevant in the current context.
- **Data abstraction** is the separation of the properties of a data type (its values and operations) from the implementation of that data type.
- **An abstract data type (or ADT)** is a programmer-defined data type that specifies a set of data values and a collection of well-defined operations that can be performed on those values.
- **Stack**

## QUESTION NO. (2) (10 MARKS) [STRING-LISTS-TUPLES-DICTIONARY]

Q2-A) List the different operations (at least three) which can be performed on (String-Lists-tuples-Dictionary).

| Data structure | Operations |
|---|---|
| String | Slicing – Concatenation - In and Not In |
| Lists | items.append(x) Add item x to the end of items.<br>items.insert(i, x) Insert item x into items at index i.<br>items.pop() Remove and return the last item in items.<br>items.pop(i) Remove and return items[i].<br>items.remove(x) Remove item x from items. Raises ValueError when x not in items.<br>items.reverse() Reverse the order of the elements in items.<br>items.sort() Sort the list items. |
| Dictionary | ➡ len(my_dict)<br>➡ element in my_dict<br>➡ my_dict.copy |
| tuples | • left hand side assignment - - In and Not In - Concatenation |

**Q2-B) Write a Python program to multiplies all the items in a list.**

```
list1=[3,1,2,3]
mul=1
for x in list1:
  mul *= x
print ('mul= ' , mul)
```

**Q2-C) Write a Python program to find maximum and the minimum value in a set.**

```
#Create a set
setA = set([5, 10, 3, 15, 2, 20])
#Find maximum value
print(max(setA))
#Find minimum value
print(min(setA))
```

**Q2-D) Write a Python program to sort (ascending and descending) a dictionary by value.**

```
import operator
d = {1: 2, 3: 4, 4: 3, 2: 1, 0: 0}
print('Original dictionary : ',d)
sorted_d = sorted(d.items(), key=operator.itemgetter(0))
print('Dictionary in ascending order by value : ',sorted_d)
sorted_d = sorted(d.items(), key=operator.itemgetter(0),reverse=True)
print('Dictionary in descending order by value : ',sorted_d)
```

## QUESTION NO. (3) (10 MARKS) [ARRAYS]

**Q3-A)** What are the Similarities and differences between the array and the list?

- Similarities: The two structures are both sequences that are composed of multiple sequential elements that can be accessed by position.
- Differences:
  - An array has a limited number of operations. which commonly include those for **array creation, reading a value from a specific element,** and **writing a value to a specific element.**
  - The list, on the other hand, provides a large number of operations for working with the contents of the list.

- The list can **grow and shrink** during execution as elements are added or removed while the size of an array cannot be changed after it has been created.
- **List is an array of pointers to python objects.**

| Arrays | Lists |
|---|---|
| Best suited for problems requiring a sequence in which the maximum number of elements are known up front. | is the better choice when the size of the sequence needs to change after it has been created. |
| Use less memory | contains more storage space than is needed to store the items currently in the list (twice the necessary capacity). |
| Faster execution | Allows for quick and easy expansion as new items are added to the list. |

**Q3-B) Define one-dimensional array and two-dimensional array.**

**One-dimensional array** is a collection of contiguous elements in which individual elements are identified by a unique integer subscript starting with zero. Once an array is created, its size cannot be changed.

A *two-dimensional array* consists of a collection of elements organized into rows and columns. Individual elements are referenced by specifying the specific row and column indices $(r, c)$, both of which start at 0.

**Q3-C) What are Modules? Why we use Modules in python? What is the ctypes and NumPy Module.**

- Modules are **files** containing Python definitions and statements (ex. *name*.py)

1. Facilitate python programs in achieving complex functions.
2. Simplicity.
3. Maintainability.
4. Code Reusability.
5. Popularity.

**NumPy**: Similar to Numeric, but handles arrays slightly differently and has a few other built-in commands and functions.

- **The ctypes Module:** This module provides access to the diverse set of data types available in the **C language** and the complete functionality provided by a wide range of **C libraries. The ctypes module** provides the capability to create hardware-

supported arrays just like the ones used to implement Python's string, list, tuple, and dictionary collection types.

**Q3-D) Write a Python program to reverse the order of the items in the array.**

```
from array import *
array_num = array('i', [1, 3, 5, 3, 7, 1, 9, 3])
print("Original array: "+str(array_num))
array_num.reverse()
print("Reverse the order of the items:")
print(str(array_num))
```

## QUESTION NO. (4) (10 MARKS) [ALGORITHM ANALYSIS]

**Q4-A)** What is the difference between Algorithms – Code – Pseudocode?

- An algorithm is a step-by-step procedure which, starting with an input instance, produces a suitable output.
- In contrast, *code* is an implementation of an algorithm that can be executed by a computer.
- Pseudocode lies between these two.

**Q4-B) Evaluate the complexity following Python Code:**

```
def ex1( n ):
  total = 0
  for i in range( n ) :
    total += i
  return total
```

$O(n)$

```
def ex3( n ):
  count = 0
  for i in range( n ) :
    for j in range( n ) :
      count += 1
  return count
```

$O(n^2)$

**Q4-C) For each function *f(n)* below, give an asymptotic upper bound using "big-Oh" notation. You should give the tightest bound possible.**

a) $f(n) = 100\,n^3 - 7n^3 + 14n^2$
b) $f(n) = 100\,n^3 - 100n^3 + 7n^2$
c) $f(n) = \log(7\,n^2)$

*a) $O(n^3)$ b) $O(n^2)$ c) $O(\log(n))$*

**QUESTION NO. (5) (20 MARKS) (DYNAMIC DATA STRUCTURES)**

**Q5-A)** Write the Pseudocode Add (value) for the Singly Linked List.

```
1) algorithm Add(value)
2)   Pre:  value is the value to add to the list
3)   Post: value has been placed at the tail of the list
4)   n ← node(value)
5)   If head = 0
6)     head ← n
7)     tail ← n
8)   else
9)     tail.Next ← n
10)    tail ← n
11)  end If
12) end Add
```

**Q5-B)** We can implement each of the stack operations with just a few lines of code. Write the Pseudocode for *STACK-EMPTY(S). PUSH(S,x). Pop(S,x).*

```
STACK-EMPTY(S)
1  If S.top == 0
2       return TRUE
3  else return FALSE

PUSH(S, x)
1  S.top = S.top + 1
2  S[S.top] = x

POP(S)
1  If STACK-EMPTY(S)
2       error "underflow"
3  else S.top = S.top - 1
4       return S[S.top + 1]
```

**Q5-C)** Suppose q is an instance of the Queue class and assume that the array implementation is used. Also, assume the size of the array is 5. Show q after all of the following operations have been completed. Assume the queue is empty to start with. Show how the front, rear, and elements change.

**(i)** q.enqueue(39); **(ii)** q.enqueue (22) ; **(iii)** item1= q. dequeue( ); **(iv)** q.enqueue(59); **(v)** item2= q. dequeue( ); **(vi)** item3= q. dequeue( )

| operation | front | rear | elements change |
|---|---|---|---|
| q.enqueue(39) | 39 | 39 | 39 |
| q.enqueue (22) | 39 | 22 | 39,22 |
| item1= q. dequeue( ); | 22 | 22 | item1= 39 |
| q.enqueue(59) | 22 | 59 | 22,59 |
| item2= q. dequeue() | 59 | 59 | item2= 22 |
| Item3= q. dequeue( ) | Null | Null | item3= 59 |

**Q5-D)** Show the stack elements after the following operations have all been completed (Draw the final picture of s.elements[ ] and show the value of s.top).
(i) s.push (20); (ii) s.push (51); (iii) s.pop(); (iv) s.push (43).

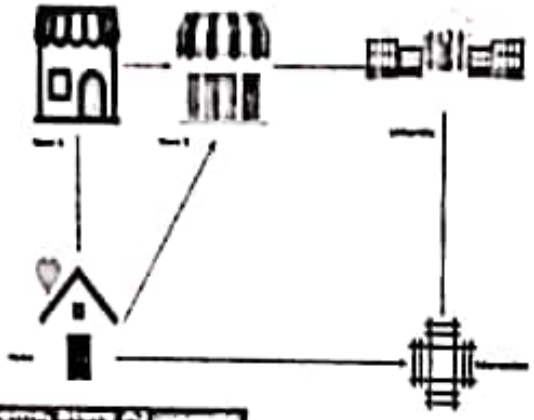
| operation | Stack elements | TOS |
|---|---|---|
| s.push (20) | 20 | 20 |
| s.push (51) | 20,51 | 51 |
| s.pop(); | 20 | 20 |
| s.push (43). | 20,43 | 43 |

**Q5-E)** What is the difference between Queue and Priority Queue?

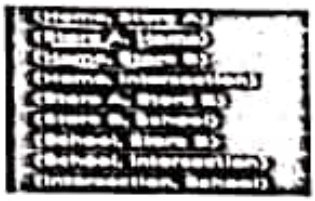
- A regular queue is a first-in and first-out data structure. Elements are appended to the end of the queue and are removed from the beginning of the queue.
- In a priority queue, elements are assigned with priorities.
- When accessing elements, the element with the highest priority is removed first.

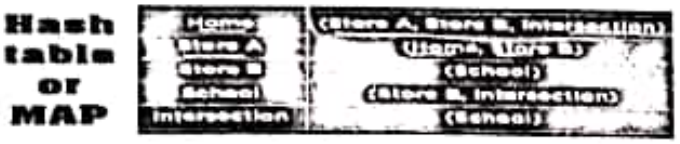
**Q5-F) For the following figure:**

- List two data structures to implement a map.
- Write the steps to find the shortest path from home to the university.
- What is the relation between data structure and the algorithm used to find the shortest path form home to the university?



| Home | (49.2, -123.4) |
|---|---|
| Store A | (49.3, -123.4) |
| Store B | (49.3, -123.3) |
| School | (49.3, -123.2) |
| Intersection | (49.2, -123.2) |





- ***Algorithms*** manipulate the data in these structures in various ways, such as: Inserting a new data item, Searching for a particular item , and Sorting the items.
- ***Using hash table or map in this case will be faster than using a list or array.***