Data Structure, Quiz 1

A. Single Choice Questions (5% each, 60%)

(1) Which of the following best defines polymorphism in the context of object-oriented programming?

(A) The ability of different objects to utilize the same interface through abstraction.

(B) The capability of a single function or method to work in different ways based on the input.

(C) The process of wrapping data and code together as a single unit.

(D) The concept of creating new classes that are derived from existing classes.

(B) or (A)

(2) In the context of OOP, what does "encapsulation" refer to?

(A) The technique of making the fields in a class private and providing access through public methods.

(B) The ability of an object to take on many forms.

(C) The process of inheriting the properties of a parent class into a child class.

(D) The composition of objects to create complex types.

(A)

(3) Which has a better worst-case time complexity between the following two algorithms? Algorithm A: $2n^3 + 100n$ Algorithm B: $50n^2 + 2n + 10$

(A) Algorithm A has a better worst-case time complexity

(B) Algorithm B has a better worst-case time complexity

(C) Both algorithms have the same worst-case time complexity

(D) It is impossible to determine without knowing the input size

(B)

(4) Which operation in a dynamic array (like Python's list) is not $O(1)$ time complexity?

(A) Accessing an element by index

(B) Appending an element to the end

(C) Inserting an element at the beginning

(D) Deleting an element from the end

(C)

(5) What is the primary advantage of using the COOrdinate (COO) format to store sparse matrices?

(A) It allows for efficient arithmetic operations.

(B) It reduces memory usage by storing only non-zero elements.

(C) It enables faster access to individual elements.

(D) It simplifies the implementation of matrix operations.

(B)

(6) Given a two-dimensional array `A` with dimensions 5x5, stored in column-major order, what is the memory address difference between `A[3,3]` and `A[3,4]` if each element occupies one memory slot and the first element `A[1,1]` is at address 0?

(A) 1

(B) 5

(C) 25

(D) 20

(B)

(7) What is True in the deletion operation of a linked list?

(A) You only need access to the node to be deleted.

(B) You need access to the node before the node to be deleted.

(C) You can delete a node without any reference to other nodes.

(D) Deletion of the last node cannot be performed.

(B)

(8) Identify the True statement about linked lists.

(A) The elements of a linked list are stored in contiguous memory locations.

(B) Linked lists require a fixed size to be declared at the time of list creation.

(C) Linked lists use more memory per element than array due to storing references along with data.

(D) Accessing an element by its index in a linked list is faster than an array.

(C)

(9) Consider an algorithm that requires the retrieval of elements in the same order they were stored, but also needs efficient insertion and deletion from both ends. Which data structure is most suitable?

(A) Stack

(B) Queue

(C) Deque

(D) Array

(C)

(10) What is the result of evaluating the prefix expression `*+AB-CD`?

(A) `(A+B) * (C-D)`

(B) `A + (B * (C-D))`

(C) `(A + B) * C - D`

(D) `A + B * C - D`

(A)

(11) How can a recursive function be prevented from running into an infinite loop?

(A) By ensuring the condition of a loop used inside the function.

(B) By including a base case that stops the recursion.

(C) By calling the function with the same arguments every time.

(D) By limiting the function to only have limited recursive calls.

(B)

(12) Which of the following statements is TRUE?

(A) Recursive functions use less memory than iterative functions because they complete tasks faster.

(B) A recursive function may have multiple base cases to handle different stopping conditions.

(C) The definition of recursion is a function that calls another function to complete its task.

(D) To solve the Tower of Hanoi problem for three disks, we need three recursive calls.

(B)

B. Short-answer questions, Please provide the derivation for each question along with your answer (8% each, 40%).

(13) Briefly explain each of the following terms.

(a) Inheritance

(b) Referential array

(c) Big-O

(d) Ordered Linked List

(a) Inheritance: In object-oriented programming, inheritance allows one class (known as the subclass) to acquire the properties and behaviors (methods) of another class (the superclass), facilitating code reuse and the creation of a hierarchical classification of classes. This mechanism supports the concept of "is-a" relationships, where the subclass is a specialized version of the superclass.

(b) Referential Array: A referential array is used in programming to store references (or pointers) to objects instead of directly storing the objects themselves. Each element in the array points to an object, allowing for more dynamic data structures since the objects can vary in size and type. This approach is beneficial when working with complex objects in language linke `Python`.

(c) Big-O Notation: Big-O notation is a theoretical measure used to describe the efficiency of an algorithm in terms of time (execution time) as a function of the size of the input data. It provides an upper bound on the growth rate of the algorithm's resource requirement, allowing for the comparison of the inherent scalability of different algorithms.

(d) Ordered Linked List: An ordered linked list is a variation of the linked list data structure where the elements are automatically arranged in a specific order (e.g., ascending or descending) as they are added to the list. Unlike a standard linked list, where insertion is typically at the head, in an ordered linked list, each new element is inserted at a position that

maintains the list's order. This property makes ordered linked lists suitable for applications requiring sorted data.

(14-1) Evaluate the time complexity of the following code segment in terms of Big-O notation:

```
for i in range(n):
    for j in range(i, n+1):
        for k in range(i, j+1):
            sum += 1

for p in range(n*n):
    for q in range(p):
        sum -= 1
```

$$\sum_{i=0}^{n-1}\sum_{j=i}^{n}\sum_{k=i}^{j}1=\sum_{i=0}^{n-1}\sum_{j=i}^{n}(j-i+1)=\sum_{i=0}^{n-1}\frac{(n-i+1)(n-i+2)}{2}=O(n^3)$$

$$\sum_{p=0}^{n^2}\sum_{q=0}^{p-1}1=\sum_{p=0}^{n^2-1}p=\frac{(n^2-1)(n^2)}{2}=O(n^4)$$

So the total Big-O is $O(n^4)$

(14-2) Evaluate the time complexity of the following code segment in terms of Big-O notation:

```
for(int i=0; i<n; i++){
    for(int j=i; j<=n; j++){
        for(int k=i; k<=j; k++){
            sum++;
        }
    }
}

for(int p=0; p<n*n; p++){
    for(int q=0; q<p; q++){
        sum--;
    }
}
```

$$\sum_{i=0}^{n-1}\sum_{j=i}^{n}\sum_{k=i}^{j}1=\sum_{i=0}^{n-1}\sum_{j=i}^{n}(j-i+1)=\sum_{i=0}^{n-1}\frac{(n-i+1)(n-i+2)}{2}=O(n^3)$$

$$\sum_{p=0}^{n^2}\sum_{q=0}^{p-1}1=\sum_{p=0}^{n^2-1}p=\frac{(n^2-1)(n^2)}{2}=O(n^4)$$

So the total Big-O is $O(n^4)$

(15) Considering the conversion of the infix expression `4+3*(6*3-9)` to its postfix form

(a) What is the first operator that appears in the postfix expression?

(b) When using a stack for this conversion, as described in our class, what is the maximum number of symbols that can be present in the stack at any moment during the conversion?

(a) The postfix representation is `4363*9-*+`, so the first operator is `*`.

(b) 4 and the stack is as follows where the right hand side is the top of the stack.

`[]`

`[+]`

`[+]`

`[+,*]`

`[+,*,(]`

`[+,*,(]`

`[+,*,(,*]`

`[+,*,(,*]`

`[+,*,(,-]`

`[+,*,(,-]`

`[+,*]`

`[]`

(16) Assuming that we have a queue Q that implements the operations defined in queue ADT. What will be the content of the queue after the following operation and why? Be sure to point out the front and rear of the queue in your answer.

```
Q = queue()
Q.enqueue(6)
Q.enqueue(20)
Q.dequeue()
if (Q.is_empty()) Q.dequeue()
Q.enqueue(2)
Q.enqueue(5)
```

After the first step: [6]

After the second step: [6,20]

After the third step: [20]

After the fourth step: [20]

After the fifth step: [20,2]

After the sixth step: [20,2,5]

The content will be [20,2,5] where 20 is in the front and 5 is in the rear.

(17) Suppose that `A[1][0]` is located at address 8, `A[3][3]` is located at address 144, `A[4][1]` is located at address 184. What is the address of `A[5][5]`? Is it column-major or row-major? What is the size of each element?

Assume `A[i][j]` = $x + \alpha * i + \beta * j$

We have

$$8 = \alpha * 1 + \beta * 0 + x$$

$$144 = \alpha * 3 + \beta * 3 + x$$

$$184 = \alpha * 4 + \beta * 1 + x$$

$\alpha = 56, \beta = 8, x = -48$

`A[5][5]` = $56 * 5 + 8 * 5 - 48 = 272$

It is row major layout since $\alpha > \beta$ and the size of element is 8.