**Topic 15: Search and sort techniques**

**Formative Assessment 15**

**Search and sort techniques (115373)**

*Hello and welcome to the assessment. Here you’ll prove to the world just how much you know and understand about what you’ve just learnt in the learner guides. This is an important part of your time at Umuzi because once this is done, you’ll be certified! So please, take this time to learn everything you can! Take a look at some pointers below with regard to answering the questions…*

* *Be specific*
* *Write professionally - no shorthand!*
* *Your answers must be original and come from your brain and your brain only.*
* *No copy/paste tricks! Our markers have seen it all and will know if you’re taking shortcuts.*
* *Remember, sloppy or poor work will be sent back to you to do again, so do it properly the first time and you’ll be done in no time.*
* *Ask for help at any time. Ask your friends, a manager, anybody!!*
* *Don’t skip any questions! You must do them all!*
* *You’ll see two boxes after each question - one for your answer and one for the marker’s comments. DO NOT delete the marker’s comments if you are required to resubmit your work after the first attempt. Should you have to do it again you will see a new box* ***under*** *the marker’s comments, so fill that one out in* ***PURPLE****. Remember!! It’s not the end of the world if you have to resubmit. You’re here to learn, so don’t beat yourself up if you don’t get it right on the first go. Obviously, try your best to get it right on the first attempt, but if not, you have another chance to do it properly!*

*Ok, and that’s that! Time to get to it! Good luck, have fun and enjoy! :)*

**Enter your name and surname below**

|  |
| --- |
| **Luvo Spofana** |

**1.** **Explain what an abstract data type is and why we make use of them? [6 Marks] (6 SO:1 AC:1)**

**Your answer below**

|  |
| --- |
| A mathematical model of the data objects that make up a data type as well as the functions that operate on these objects is referred to as an Abstract data type. There are no standard conventions for defining abstract data types.  They are used to manage the complexity of problems and the problem-solving process; computer scientists use abstractions to allow them to focus on the “big picture” without getting lost in the details. By creating models of a problem, we can utilize a better and more efficient problem-solving process. |

**Marker’s Comments**

|  |
| --- |
|  |

**2. Describe three characteristics of data structures [3 Marks](3 SO:1 AC:2)**

**Your answer below**

|  |
| --- |
| Correctness − Data structure implementation should implement its interface correctly.  Time Complexity − Running time or the execution time of operations of data structure must be as small as possible.  Space Complexity − Memory usage of a data structure operation should be as little as possible. |

**Marker’s Comments**

|  |
| --- |
|  |

**3. Explain the differences between bubble sort and insertion sort? [5 Marks](5 SO:2 AC:1-2)**

**Your answer below**

|  |
| --- |
| Bubble sort is a simple sorting algorithm that works by comparing the first pair of items in the array and swapping them if they are in the wrong order. The algorithm continues with the next pair of items in the array, then the next, and the next, until it reaches the end. Then the algorithm goes back to the beginning and repeats this process until every item is in the correct order.  Insertion sort is a comparison-based algorithm that builds a final sorted array one element at a time. It goes through an array one element at a time and inserts that element into the right place. By the time the algorithm gets to the last element in the array, all the elements will be sorted. It is slightly faster than bubble sort (and selection sort) because it doesn’t have to loop through the array as many times. |

**Marker’s Comments**

|  |
| --- |
|  |

**4. Explain the differences between linear search and binary search? [5 Marks](5 SO:3 AC:1-2)**

**Your answer below**

|  |
| --- |
| * The linear search starts searching from the first element and compares each element with a searched element till the element is not found, while a binary search finds the position of the searched element by finding the middle element of the array. * In a linear search, the elements don't need to be arranged in sorted order. The pre-condition for the binary search is that the elements must be arranged in a sorted order. * A linear search can be implemented on both a single and multidimensional array. A binary search can be implemented only on a multidimensional array. |

**Marker’s Comments**

|  |
| --- |
|  |

**5. A) List and explain the typical problems encountered when using sorting and searching techniques?**

**B) Explain how “Big O” notation helps programmers think about these problems? [8 Marks](8 SO:2-3 AC:3)**

|  |
| --- |
| **A)**  The problem with sorting techniques is taking items in a list and rearranging them in an order that is nondecreasing, and this issue becomes meaningful when the nature of the given list’s items must allow for the items to be sorted in such a manner. Generally, lists consist of different items, and for them to be able to be sorted, a piece of information about each item is taken to guide the sorting, such a specially chosen piece of information is called a key.  The problem with the searching techniques deals with finding a search key, which is a given value in each set/multiset that allows several elements to have the same value.  There is no single algorithm that fits all situations best. Some algorithms work faster than others but require more memory; some are very fast but applicable only to sorted arrays; and so on.  Searching algorithms encounter different issues in applications where the underlying data may frequently change, relative to the number of searches, because searching then requires to be considered in conjunction with 2 other operations: adding to, and deleting to the data set of an item. In such situations, careful consideration of algorithms is required to strike a balance among the requirements of each operation.  **B)**  Big O is often used to describe the worst-case of an algorithm by taking the highest order of a polynomial function and ignoring all the constants value since they aren’t too influential for sufficiently large input. Big-O tells you the complexity of an algorithm in terms of the size of its inputs. This is essential if you want to know how algorithms will scale. ultimately, Big-O notation helps you determine which algorithms are fast, which are slow, and the tradeoffs. |

**Marker’s Comments**

|  |
| --- |
|  |