**Name:**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ SID **#:**\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Matching** (6 points each)

*For each term in the left column, write the letter for the description that best matches the term from the right column.*

\_\_\_\_ 1. Stack

\_\_\_\_ 2. Hash function

\_\_\_\_ 3. Linked list

\_\_\_\_ 4. Queue

\_\_\_\_ 5. Load factor

\_\_\_\_ 6. Priority queue

1. The number of records in the hash table divided by the size of the hash table.
2. Linear collection of self-referential class objects called nodes connected by reference links.
3. First In First Out data structure.
4. Customers or jobs with higher priority are pushed to the front of the data structure.
5. Last In First Out data structure.
6. Takes a search key and produces the integer index of an element in the hash table.

**True/False** (6 points each)

*Indicate whether the sentence or statement is true or false.*

\_\_\_\_7. f(n) = 4n + 2n2 + 5 ∈ O(n).

\_\_\_\_8. A sequential search of a list assumes that the list is in ascending order.

\_\_\_\_9. Binary search can be performed on both sorted and unsorted lists

**Multiple Choice** (6 points each)

*Identify the letter of the choice that best completes the statement or answers the question.*

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | [0] | [1] | [2] | [3] | [4] | [5] | [6] | [7] | [8] |
| list | B | D | F | H | L | N | P | R | U |

\_\_\_\_10. Using binary search algorithm, how many key comparisons would have to be made on the list above to find the letter P?

|  |  |  |  |
| --- | --- | --- | --- |
| a. | 7 | c. | 6 |
| b. | 2 | d. | 5 |

**Short-Answer questions** (10 points each)

lastNode

*Answer the following questions in the space provided.*

45

65

76

34

firstNode

newNode

25

1. Assume the above setting of a linked list. Write the Java code to insert the new node referenced by *newNode* into the list referenced by *firstNode* at the front of the list.

25

45

65

76

34

lastNode

firstNode

newNode

1. Assume the above setting of a linked list. Write the Java code to insert the new node referenced by *newNode* into the list referenced by *firstNod*e between nodes whose values are 65 and 34.

firstNode

lastNode

34

76

65

45

1. Assume the above setting of a linked list. Write the Java code to delete the node whose value is 34 from the list.

86

22

lastNode

firstNode

30

12

. . .

1. Assume the above setting of a linked list with *n* number of nodes. Write the Java code to count how many nodes are in the list.