This exam is closed book, closed notes. Don’t refer to any other materials other than the exam itself. Anyone caught cheating faces disciplinary action, which could result in expulsion from the University.

Write your name on every page, including reference and scratch paper. Scratch paper must be turned in at the end of the exam.

You have 3 hours to complete this exam. Work to maximize points. If you don’t know the answer to a problem, move on and come back later. Most importantly, stay calm and don’t panic. You can do this.

Name: _________________________________
ID: _________________________________    Proctor Initial: _________________________________

If there is a student to your immediate left or right, write his/her name in the space provided

Name of student to your LEFT: _________________________________    Name of student to your RIGHT: _________________________________

(Write “N/A” if seat immediately to your left or right is not occupied, or a wall or aisle, etc.)

DO NOT OPEN THIS EXAM UNTIL YOU ARE INSTRUCTED TO DO SO.

GOOD LUCK!
Part 1: The Basics [40 points]
This section tests your basic knowledge of data structures via multiple choice questions. Circle the most accurate answer. Circle only one option, choosing multiple options will result in zero points. [2 points per questions]

1. What is the maximum number of children of a node in a binary search tree?
   A. 0
   B. 1
   C. 2
   D. 3

2. What is the minimum number of leaves in a binary search tree with N (N > 1) keys?
   E. \((N + 1)/2\)
   F. \(N - 1\)
   G. 0
   H. 1

3. Consider a Huffman tree to encode M distinct symbols with non-zero frequencies. How many leaves are present in the Huffman tree?
   A. 2
   B. \(\log_2(M)\)
   C. M
   D. \(2^M\)

4. What is the worst case height of a multiway trie storing N elements, each comprised of at most D-digits?
   A. \(O(\log(N))\)
   B. \(O(\log(D))\)
   C. \(O(D)\)
   D. \(O(N)\)

5. One of the goal of AVL trees is to maintain the difference between height of left and right children to within +/-1. Which of the following helps maintain the data structure?
   A. Left rotation
   B. Right rotation
   C. Both left and right rotation
   D. None of the above

6. What is the minimum number of edges in an undirected connected graph with N nodes?
   A. 2
   B. N
   C. \(N - 1\)
   D. \(N/2\)
7. Which of the following algorithms finds the shortest path from a source vertex to a destination vertex in an un-weighted undirected connected graph? Choose the best answer.
   A. Depth First Search (DFS)
   B. Breadth First Search (BFS)
   C. Dijkstra’s algorithm
   D. Both BFS and Dijkstra’s algorithm

8. Which of the following algorithms is best suited for finding a minimum spanning tree? Choose the best answer.
   A. Depth First Search (DFS)
   B. Breadth First Search (BFS)
   C. Dijkstra’s algorithm
   D. Prim’s algorithm

9. What is the difference in the number of edges between the minimum spanning tree returned by Prim’s algorithm and Kruskal’s algorithm when run on the same graph? Assume all the edge weights of the graph are distinct.
   A. -1
   B. 0
   C. 1
   D. 2

10. What is the minimum number of black nodes in a red-black tree with exactly three nodes?
    A. 0
    B. 1
    C. 2
    D. 3

11. Which of the following is TRUE about a skip list with N elements?
    A. The elements in the skip lists are in a sorted order.
    B. Worst case run time of find operation is O(log(N)).
    C. Every node has equal number of forward pointers.
    D. Worst case run time of inserting an element is O(1).

12. Which of the following is FALSE about a red-black tree?
    A. Root is black
    B. Every subtree of a red-black tree is a valid red-black tree
    C. Every node is either red or black
    D. A red node can’t have a red child
13. What is the worst-case run time of deleting the minimum element in a min-heap with \( N \) elements?
   A. \( \Theta(1) \)
   B. \( \Theta(\log N) \)
   C. \( \Theta(N) \)
   D. \( \Theta(N \log N) \)

14. Which of the following operations uses path compression in a union-find data structure?
   A. Union
   B. Find
   C. Create
   D. Both Union and Find

15. Which is the following data structure have the best worst-case run time of inserting an element?
   A. Sorted array
   B. Skip list
   C. Simple binary search tree
   D. Red black tree

16. Which of the following is FALSE about a B-Tree?
   A. A leaf in a B-tree can have twice as many elements as another leaf in the same B-tree.
   B. B-trees are balanced search trees.
   C. B-Tree is a binary tree
   D. A node in a B-tree is designed to fit in one disk block.

17. If we have no rule in place for performing the union of two up trees, what is the worst case run time of find and union operations in terms of the number of elements \( N \)?
   A. find: \( \Theta(N) \), union: \( \Theta(1) \)
   B. find: \( \Theta(1) \), union: \( \Theta(N) \)
   C. find: \( \Theta(\log 2N) \), union: \( \Theta(1) \)
   D. find: \( \Theta(1) \), union: \( \Theta(\log 2N) \)

18. Which of the following strategy will reduce the runtime complexity of union operation compared to simple union and find operation?
   A. Union by height and simple find
   B. Union by size and simple find
   C. Both A and B
   D. None of the above
19. Suppose you have B+ tree data structure with $M=4$, $L=3$. Starting with an empty tree you inserted following sequence of numbers: 12, 13, 4, 8, 1, 15, 18, and 19. How many leaves will be present in the final B+ tree?

   A. 1  
   B. 2  
   C. 3  
   D. 4

20. Which of the following is FALSE about a 2-3 tree?

   A. All the leaves are at the same level.  
   B. The root node must be a 2-node.  
   C. Every node has either one or two data elements.  
   D. If an internal node have one data element, then it must have two children.
Part 2: Application and Comparison [20 points]

This section goes under the hood of the data structures covered in class, their strengths and weaknesses and applications. The format is short answers and fill in the blanks.

1. A binary search tree is built as shown below after inserting following keys in to an empty tree:
   6, 9, 4, 2, 10, 12, 7, 8
   Some of the keys are denoted as alphabet (A – E) in the binary search tree.

   ![Binary Search Tree Diagram]

   Write down the possible values of the following keys: [1 pt each]
   A. __________
   B. __________
   C. __________
   D. __________
   E. __________
2. Consider the red-black tree shown below which was built after inserting following sequence of keys in that order starting with an empty tree:
   6, 7, 10, 2, 5, 11

   For each of the nodes below write down their key value and the color [1 pt each]

   ![Red-black tree diagram]

<table>
<thead>
<tr>
<th>Nodes</th>
<th>Key</th>
<th>Color</th>
</tr>
</thead>
<tbody>
<tr>
<td>F</td>
<td>11</td>
<td>red</td>
</tr>
<tr>
<td>A</td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td></td>
<td></td>
</tr>
<tr>
<td>C</td>
<td></td>
<td></td>
</tr>
<tr>
<td>D</td>
<td></td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
3. Consider the following set of initially unrelated elements $S = 0, 1, 2, 3, 4, 5$. Assume that the initial union-find data structure is represented by an up-tree consisting of a forest of 6 singleton nodes. Assume union-by-height for all of the following questions.

A. Draw the array representation of the uptree for the 6 singleton nodes [1 pt]

```
 0 1 2 3 4 5
```

B. Draw the final contents of the array representing the uptree after the following sequence of operations. Note: Break ties by keeping the first argument as the root. [4 pt]

   i. $\text{Union}(0,1)$
   ii. $\text{Union}(0,2)$
   iii. $\text{Union}(3,4)$
   iv. $\text{Union}(0,3)$

```
 0 1 2 3 4 5
```
4. Kruskal’s algorithm [2 pts]
Consider following graph with given edge weights. Edges are identified by unordered pairs of nodes such as (A, B), (B, F), and (C, D). Write the CORRECT sequence of edges in the order discovered by Kruskal’s algorithm to build a minimum spanning tree in the following graph. The sequence order must be correct to get full marks.

![Graph Image]

5. Hashing [3 points]
A. A hash table of length 7 uses open addressing with hash function \( h(k) = k \mod 7 \), and linear probing. For each of the following keys compute the hash function \( h(k) \) [1 pt]:

<table>
<thead>
<tr>
<th>k</th>
<th>12</th>
<th>14</th>
<th>16</th>
<th>33</th>
<th>40</th>
<th>41</th>
</tr>
</thead>
<tbody>
<tr>
<td>( h(k) = k \mod 7 )</td>
<td>5</td>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

B. After inserting above 6 keys into an empty hash table, the table is as shown below.

<table>
<thead>
<tr>
<th></th>
<th>0</th>
<th>1</th>
<th>2</th>
<th>3</th>
<th>4</th>
<th>5</th>
<th>6</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>14</td>
<td>41</td>
<td>16</td>
<td>12</td>
<td>40</td>
<td>33</td>
<td></td>
</tr>
</tbody>
</table>

Write one possible order of insertion insertions in an initially empty hash table that can generate the above table? [2 pt]
Part 3: Run Time Analysis [20 points]

This section focuses on the algorithms covered in class. You are asked to simulate fast implementations and analyze the performance of these algorithms. The format is short answers and fill in the blanks.

1. What is the worst case run time of inserting an element in to a binary search tree containing N elements?[1 pt]

2. What is the worst case run time of finding an element in a linked list containing N elements?[1 pt]

3. What is the worst case run time of finding an element in a red-black tree containing N elements?[1 pt]

4. What is the worst case run time of find operation in a union-find data structure implementing union-by-height containing N elements, without path compression?[1 pt]

5. What is the worst case run time of finding a key in a hash table of size M with separate chaining containing N keys?[1 pt]

6. What is the runtime complexity of finding a shortest path between a given source vertex and destination vertex in an undirected unweighted connected graph with m vertices and n edges? [1 pt]

7. What is the tightest upper bound on the height of the B-tree with order m and N keys?[1 pt]

8. What is the maximum number of keys that can be stored in the root node of a B+ tree with M =4, and L=3?[1 pt]
9. Assume the input is an undirected connected graph $G$ which has $|E|$ edges and $|V|$ vertices. Support graph $M$ is the minimum spanning tree of graph $G$ and it is already computed. What would be the run time complexity of Dijkstra’s algorithm on the graph $M$? Do not include the complexity of building a minimum spanning tree. Circle the best answer (tightest bound). [2pts]

a. $O(|V|)$
b. $O(|V| \log(|V|))$
c. $O(|E|)$
d. $O(|V|^2)$

For the following two questions, consider the weighted undirected graph $G=(V,E)$ defined by the adjacency matrix below where weights represent the distance between the nodes.

<table>
<thead>
<tr>
<th></th>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td></td>
<td>1</td>
<td>6</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>B</td>
<td>1</td>
<td></td>
<td>5</td>
<td>4</td>
<td></td>
<td>3</td>
</tr>
<tr>
<td>C</td>
<td>6</td>
<td>5</td>
<td></td>
<td></td>
<td>8</td>
<td></td>
</tr>
<tr>
<td>D</td>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>E</td>
<td></td>
<td>8</td>
<td></td>
<td></td>
<td>7</td>
<td></td>
</tr>
<tr>
<td>F</td>
<td>3</td>
<td>2</td>
<td>7</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

10. A breadth-first search (BFS) starting from vertex A will visit the 6 vertices in $G$ in some sequence. Write the sequence of vertices visited by BFS if edge weights are used to order the sequence of vertices when multiple choices are possible. In other words, when multiple vertices may be visited from a given vertex, choose to first visit the one connected via the minimum edge weight. [5 pt]

11. Write the sequence of edges that would be part of the minimum spanning tree returned by Prim’s algorithm starting from vertex A. Indicate edges as a ordered pair of vertices in the sequence accepted by Prim’s algorithm.[5 pt]
Part 4: C++ and Programming Assignments [20 pts]

This section tests your understanding of C++ and the programming assignments. The format is short coding problems and multiple choice.

   Circle the line number where there is an error in the following program.

   ```
   int main() {
   int a = 0;
   int& b;
   b = a;
   }
   ```

13. C++ stl container. [2pt]
   Write the sequence of numbers printed in the STDOUT after running the following program? Assume priority_queue implements max heap data structure.

   ```
   int main() {
   priority_queue<int> a;

   for (int i = 1; i <= 3; ++i) {
       a.push(i);
   }

   for (int i = 0; i < 3; ++i) {
       cout << a.top() << ' ';
       a.pop();
   }
   cout << endl;
   return 0;
   }
   ```
14. **PA1**: Recall your implementation of the class BST which represents a binary search tree. The BSTIterator increment operator updates the BSTNode pointer to the inorder successor of the current BSTNode. We are planning to test your implementation using following code. What is the sequence of numbers printed to STDOUT? [2 pts]

```cpp
BST<int> bst;
for(int i : {1,8,5,6}) {
    bst.insert(i);
}
for (auto it = bst.begin(); it != bst.end(); ++it) {
    cout << *it << " ";
}
cout << endl;
```

15. **PA2**: Recall your implementation of inserting a new key into an RST. Assume that you are inserting a new key into following RST where letters are keys and numbers are priorities. You are planning to insert key F. The random number generator returned 35 as the priority. Draw the final RST after the insertion. [3 pt]

![Diagram of RST](image)

A
5
e
6
L
11
D
10
J
15
G
30
16. PA3: Recall your implementation of the build method that constructed the Huffman Trie from a vector of symbol frequency counts. Assume that the class HCTree represents the Huffman Trie, and the class HCNode represents a node in the trie. Following is a definition of the operator < overloading on the HCNode object. [5 pts]

```cpp
bool HCNode::operator<(const HCNode& other) const {
    if (count == other.count) return symbol > other.symbol;
    else return count > other.count;
}
```

Consider this context:

HCNode n1, n2, n3, n4;
```
n1.count = 45; n1.symbol = 'C';
n2.count = 98; n2.symbol = 'B';
n3.count = 45; n3.symbol = 'A';
n4.count = 45; n4.symbol = 'C';
```

e. What is the value of following expressions? Circle True or False. [0.5 pt each]
   
i. n1 < n2 - True  False
   ii. n2 < n3 - True  False
   iii. n1 < n3 - True  False
   iv. n1 < n4 - True  False

f. Bitwise operation: Suppose we want to read the \( n^{th} \) bit from the right of a byte argument \( b \), and return the result in a int (0 or 1 ). Fill in the blanks in the code below for the function readBit. The rightmost bit in the byte corresponds to \( n=0 \) [3 pt]

```cpp
int readBit(char b, int n) {
    int c = (_______<< n);
    int d = (_______& c);
    return (_______>> n);
}
```
17. PA4: Recall your implementation of `buildLexicon` that takes as argument a set containing the words specifying the official lexicon to be used for the game, and `isInLexicon` that takes as argument a const string passed by reference, and determines whether it can be found in the lexicon. [6pts]

a. If the lexicon contains N words with at most D-characters, what is the runtime complexity of `buildLexicon` assuming multiway trie implementation? [2pt]
   i. O(1)
   ii. O(D)
   iii. O(N * D)
   iv. O(D \log(N))

b. If the lexicon contains N words with at most D-characters, what is the runtime complexity of `isInLexicon` assuming multiway trie implementation? [2pt]
   i. O(1)
   ii. O(D)
   iii. O(N)
   iv. O(\log(N))

c. If the lexicon contains N words with at most D-characters, what is the runtime complexity of `buildLexicon` assuming simple binary search tree implementation? [2pt]
   i. O(N)
   ii. O(N * D)
   iii. O(N * N * D)
   iv. O(N \log(N))
Name: _________________________________

Scratch paper