Write your name in box below and NetID to the right ===>	netid
	1pm Lecture ☐
	2pm Lecture ☐
Your name goes here	3pm Lecture ☑

I wish you all good skill! Read these instructions FIRST!

Please put your name and NetID neatly on this page and your name on the rest of the pages. I unstaple these to feed them into the scanner and I have (at least once!) dropped the stack of exams and had to put them back together…

Don't use a light pencil – If the scanner can't see it, I can't grade it. If unsure, ask a proctor.

Please be smart in your time management. This is an assessment.. I want to understand your level of understanding. Don't get stuck on a question. Do the questions you are most sure about first.

Be concise, but be clear and complete. If you make assumptions, write them out. If you need extra space, there are two "scratch" pages at the end of the exam. Please tell me to look there for a continuation of your answer.

Code quality rules still apply (except for comments). So your code must be reasonably performant (e.g. O(1) where it should be O(1)), not overly complex, use good names, const, etc…

Show your work, it helps me give you partial credit even if you get lost. I really hate taking off all the points and writing "No attempt made"

You will have until the posted end time to complete the exam.

Desks must be clear. No talking. No notes or electronic devices will be allowed other than a calculator (you shouldn't need one). Don't take pictures of the exam.

If you need to use the restroom, just go – no need to ask. We're adults and we should operate from a level of trust.

For all code, you may assume that we are `using namespace std;` and that we've included all the expected files (<iostream>, <stdexcept>, <string>, <vector>, etc…)

If the exam is not on the front table in the box when I call time, then you will receive a zero. Please understand that I truly mean this – I will not tolerate anyone trying to extend the exam. It is not fair to those students who finish the exam on time.

Name: _____

1) For each of the following, write a clear definition

a) B-Tree (you can use 2-3 if you choose)

b) Big O

2a) What is definition of and the properties of a Queue

2b) What is the one required attribute of a hash function? What are the desired ones?

Name: _____

3) Write the rest of quick partition

```
void quicksort(vector<int>& A) {
  if (A.size() < 2) return;
  quicksort(A,0,A.size()-1);
}
void quicksort(vector<int>& A, int lo, int hi) {
  if (lo >= hi or lo < 0) return;
  int end_of_low = QPartition(A, lo, hi);
  quicksort(A,lo,end_of_low);
  quicksort(A,end_of_low+1,hi);
}

int QPartition(vector<int>& A, int lo, int hi) {

  int mid =                                          // fill in code

  int pivot = A[mid];
  while(true) {
    // Fill in code here









    if (lo >= hi) {
      break;
    } else { // Fill in code below









    }
  }
  return hi;
}
```

Name: _____

4) For each of the following, tell me the Big O and why you choose that value. You will need to describe each of the significant parts of the algorithm in sufficient detail to account for the work. Detail what you are counting. Don't be vague (like only saying "height of the tree!")

a) Quicksort

b) Linked list reverse

c) Queue push (array)

d) Build a Heap from n inputs

Name: _____

5) Write the code to the prototype for the function I describe below. It should be suitable for insertion into a header file. The implementation will live elsewhere, you are not expected to write it.

- A function that reads all integers in a named file storing them in a vector I provide. It also returns the largest one as the function result.

6) Why do we have the rule of three?

7) Why (and how) do we check input files **and** extracted values (input >> x) for validity?

8) Describe (in pseudo code) an optimal algorithm to find the height of a heap of size n. Please give its Big O and justify it.

Name: _____

9) Complete this class with the 5 standard functions. You are responsible for writing the appropriate const and non-const methods. Do not change the private variables. You do NOT need to conform to the rule of three. I would like code for a Queue of strings. MAXSIZE is fixed. Resize is **NOT** allowed.

Do not write this where you shift the values to make index 0 the front. That makes pop O(n) instead of O(1). I will give you a zero.

```
class Queue {
  vector<string> array;
  static const int MAXSIZE = 10;  // Will not change
  int first = 0;
  int size = 0;
public:
  Queue() : array(MAXSIZE) {}
```

Name: _____

10) Complete the `HashTable::resize()` function. It is called from the `insert` function when the provided `table_looks_full()` function decides the table has too many collisions. The insert uses a helper function that you don't need to write. It takes care of collisions. The hash table does not have a remove function, so can we use empty strings to indicate empty slots in the table. The hash() function returns a number from 0..MAXINT.

```
class HashTable {
  vector<string> table;
  void insert(vector<string>& array, const string& key);
  bool table_looks_full() const;
  int hash(const string& key) const;
public:
  HashTable() : table(100) {}  void insert(const string& key) {
    if (key.empty()) throw runtime_error("empty strings not allowed");
    if (table_looks_full()) resize();
    insert(table,key);
  }
  void resize() {
```

Name: _____

11) Please write **only** insert_front for this doubly linked list. Some of the routines are left out for clarity, but you have everything you need to write this insert.

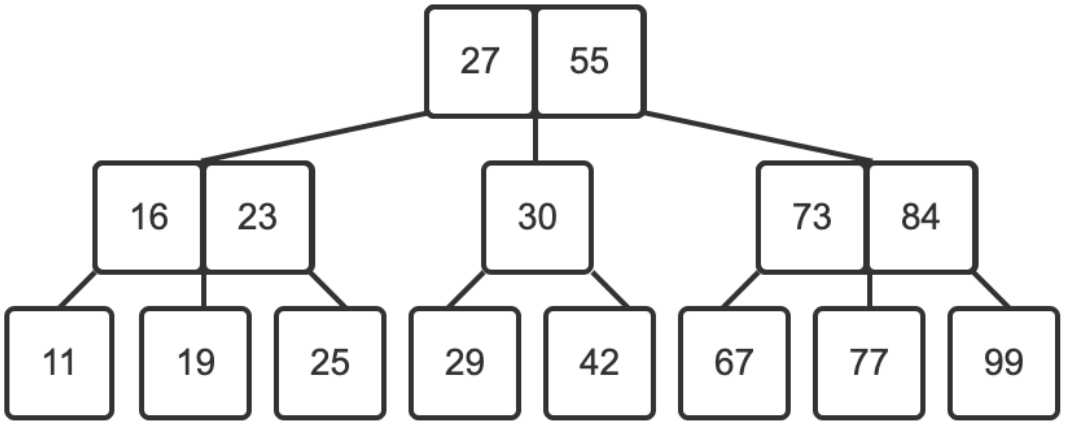
```
class DoubleLinkedList {
  struct Node {
    string key;
    Node* next;
    Node* prev;
    Node(const string& key) : key(key),next(nullptr),prev(nullptr) {}
  };

  Node* dummy_head;
  Node* dummy_tail;
public:
  DoubleLinkedList()
    : dummy_head(new Node("")),
      dummy_tail(new Node(""))
  {
    dummy_head->next = dummy_tail;
    dummy_tail->prev = dummy_head;
  }

  void insert_back(const string& key);
  void insert_front(const string& key) {
```

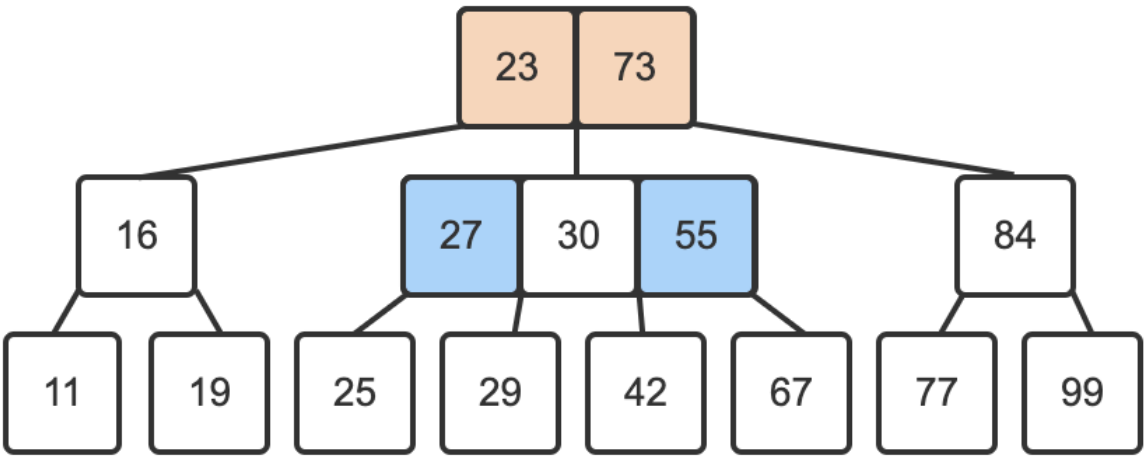
Name: _____

12) Here's a 2-3 tree. Please insert 80 and 81. Draw the resulting tree.



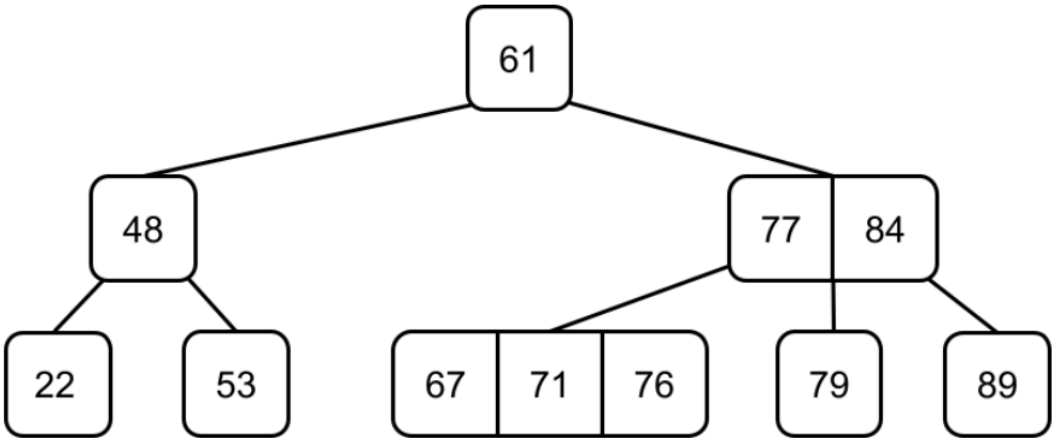
Name: _____

13) Here's a 2-3-4 tree. Please insert 68 and 70. Draw the resulting tree.



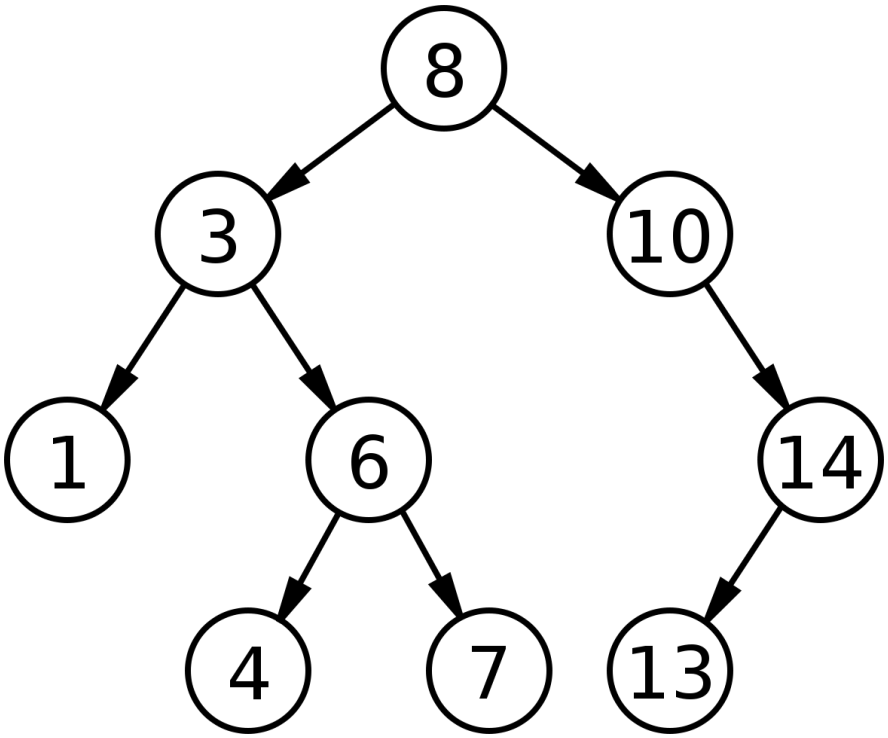
Name: _____

14. Here's a 2-3-4 tree. I want you to draw two trees. First, remove 79 and draw that tree. Then remove 89. Draw that second tree.



Name: _____

15) Here's an AVL tree. I just added the node with 13. Draw in the balance factor for each node below the key. If not valid, make the appropriate fix and draw below.



Name: _____

16a) For Red-Black trees, what are the rules

16b) Here's a Red-Black tree.

a) What color is the node labeled [8]
b) What color is the node labeled [14]
c) What color is the node labeled [10]
d) There are 8 possible color combinations for [1] [3] and [6]. If it works, write YES. If a combination doesn't work, say NO and give the reason why.

1:R 3:R 6:R –

1:R 3:R 6:B –

1:R 3:B 6:R –

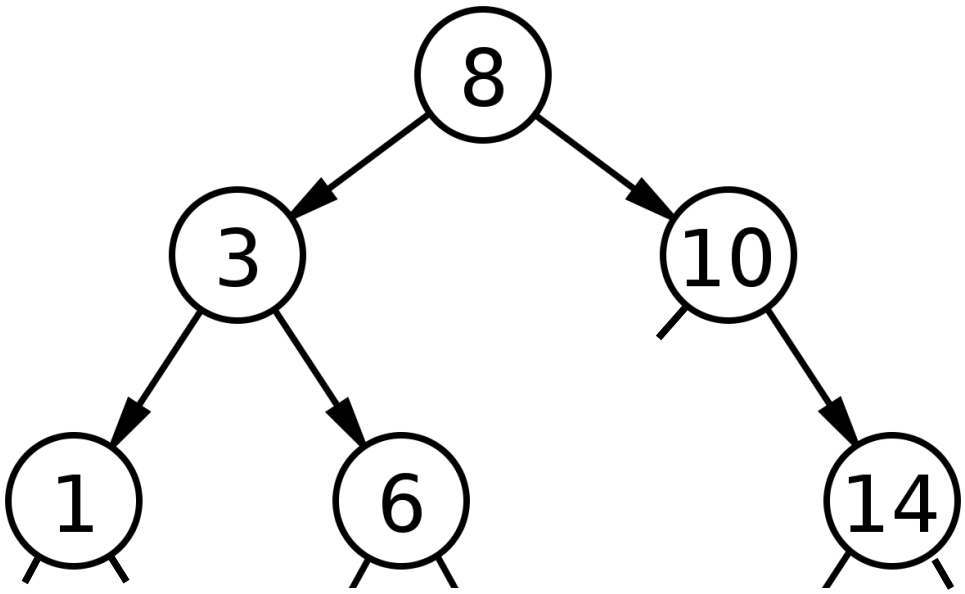
1:R 3:B 6:B –

1:B 3:R 6:R –

1:B 3:R 6:B –

1:B 3:B 6:R –

1:B 3:B 6:B –



Name: _____

17) Linux doesn't store your clear text password, but when you try to log in, the system will let you in when you type the correct one in when prompted. How does that work?

18) I made a mistake creating my hash function and it only returns even numbers. I use quadratic hashing and have 100 slots in my table. What is going to happen?

19) Insert the following 6 values (in order) into the empty hash table. Your hash function is simply the number modulo 10. The collision strategy is just quadratic probing (both of the coefficients are 1). It would be a good idea to show your work. **[64, 75, 77, 47, 55, 60]**

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

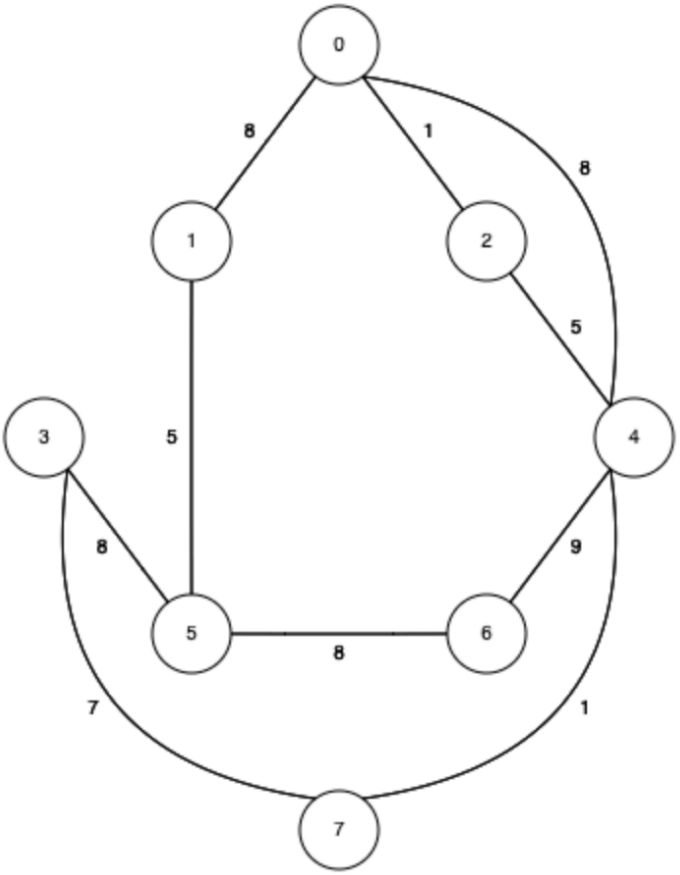
Name: _____

20) What are the steps for Dijkstra's method? Tell me what you do to set it up and what you do at each step.

21) Is Djikstra a depth first method or a breadth-first approach? How do you know?

Name: _____

22)



I want you to perform Dijkstra's Method to find all the shortest paths. Note that you are **starting from node 1** (S1 in the table). The V column is simplly there as a convenience – it is not graded. The distance and previous node entries ARE graded. You must show your work for credit here. I ask that you cross out values rather than erase or skip them. So, I expect to see things like ~~7~~ ~~5~~ 3 to indicate that the value was 7, then a distance 5 was found, and then finally 3 was found. Same for the previous nodes.

Label	V	Shortest distance found so far	Previous node in backward path
0			
S1			
2			
3			
4			
5			
6			
7			

Name: ____

Scratch 1

Name: ____

Scratch 2