Name: NetID: _____

I am normally in the ☐ 3:30 class ☐ 5 pm class

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.

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.

Code quality rules still apply (except for comments, destructors, and the rule of three). So your code must be reasonably performant (e.g. O(1) where it should be O(1)), not overly complex, use good names, 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. I know that it is Ramadan… if you need to break fast after sunset, you can step out (and I'll grant you a few extra minutes at the end)

For all code, you may assume that we are `using namespace std;` and that we've done

- `#include <iostream>`
- `#include <string>`
- `#include <vector>`
- `#include <stdexcept>`

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.

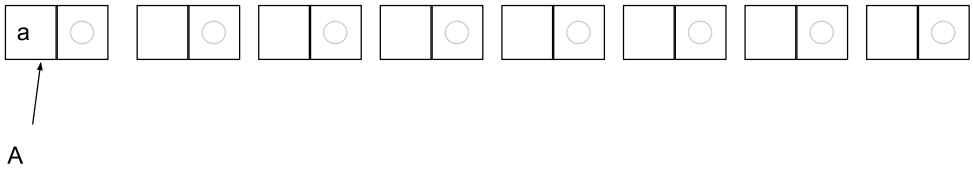
Please initial here to indicate that you understand all this and that you WILL put your name on every page [1 point for complying]: _____

Name: _____

1) Let's write a good C++ function (not just the header) that **will pass muster for a full code quality review check**. The purpose of the function is to read a file I expect to contain numbers, skip some numbers that I provide, and collect the rest. I'll want the function to accept a filename (string), a vector of numbers to ignore, so it can read and return the numbers in the file that ARE NOT in the vector of numbers to ignore.. This function definition should work for `int` and for `double` (Hint: one function definition must serve both cases). For reference, my version was about 20 lines long. There is a code overflow page you may use if needed.

Name: _____

2. Given the definition of `Node` below, draw what the memory looks after running the rest of the code like using the node boxes below. Use an arrow to indicate what variables are pointing to (e.g. if a variable named `head` points to a node, then write the word `head` and add an arrow pointing from the word `head` to the appropriate box). Use a clear and consistent indication for null pointers (leaving it blank is <u>not</u> a clear indication).




```
struct Node {
    Node(char c) : payload(c), next(nullptr) { }
    char payload;
    Node* next;
};

  Node* A = new Node('a'); // I've drawn this already
  A->next = new Node('b');
  A->next->next = A;
  A->next = new Node('c');
  Node* D = new Node('d');
  A->next->next = D;
  D->next = A->next;
  D->next->next = new Node('e');
```

Name: _____

3. Big O questions. For each of these, I first want the typical Big O. Then please describe why, in terms of the operation, it meets that Big O criteria. For example, if something is O(n), tell me what 'n' you are counting.

a) Stack::Push() [assume array based with resize]

b) Stack::Pop() [assume link list based]

c) LinkedList::append(x) [assume only a head pointer]

d) Mergesort on an array of n numbers.

e) QuickSort on an array of n numbers.

f) TwoThreeTree::insert(x)

g) AVL::insert(x)

Name: _____

4. [Big code] I want to implement a class that mimics a set of strings. I choose to do it with a binary search tree. I've provided the header. You will implement a few of these functions. If you cannot make set intersection work, this is just a BST – but you will get partial credit. For reference, my full implementation was about 50 additional lines. You may use recursive helpers if needed (add them to the header and implement them in your answer).

```
class StringSet {
  struct Node {
    string key;
    Node* left;
    Node* right;
    Node(const string& key)
        : key(key), left(nullptr), right(nullptr) {}
    ~Node() { delete left; delete right; }
    Node(const Node&) = delete;
    Node& operator=(const Node&) = delete;
  };
  Node* root;
  void copy_helper(const Node* tree) { // pre-order traversal!
    if (tree == nullptr) return;
    add(tree->key);
    copy_helper(tree->left); copy_helper(tree->right);
  }



public:
  // I'll do these for you (full rule-of-three even!)
  StringSet() : root(nullptr) {}
  ~StringSet() { delete root; }
  StringSet(const StringSet& other) : root(nullptr) {
    copy_helper(other.root);
  }
  StringSet& operator=(const StringSet& other) {
    if (this == &other) return *this;
    delete root; root = nullptr; copy_helper(other.root);
    return *this;
  }

  // Please implement these on the following two pages...
  bool is_empty() const;
  void add(const string& key);
  bool contains(const string& key) const;
  StringSet intersection(const StringSet& other) const;
};
```

Name: _____

```
bool StringSet::is_empty() const {




void StringSet::add(const string& key) {
```

Name: _____

```
bool StringSet::contains(const string& key) const {
```

```
StringSet StringSet::intersection(const StringSet& other) const {
```

Name: ____

<code overflow page>

Name: _____

5. Heaps
   a) Tell me what a heap is (definition)

   b) Take the array [97 85 14 50 82 10 2 43 81 20 75 8 6 1] and draw it as a heap tree. Is it heap?

   c) Assume I have an array A of size n that is a heap. I want to run a heap sort. What are the steps and what is the Big O. Tie the Big O value to the steps you wrote out.

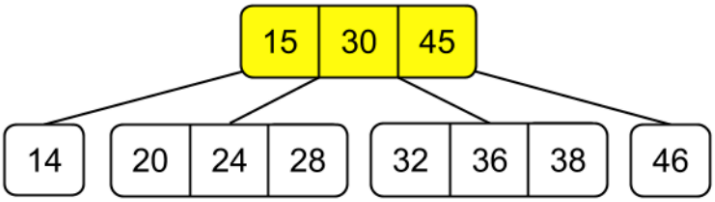
Name: _____

d) Write the code for

```
void maxheap::percolate_up(int* A, int index){
```

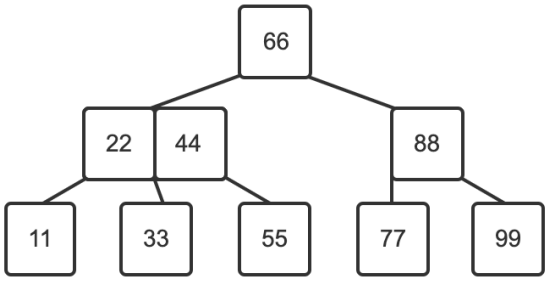
Name: _____

6. Here's a 2-3-4 tree. Draw the tree that results when you insert the key 25



Name: _____

7. Here's a 2-3 tree. Draw the tree that results when you insert 5 and 10.



Name: _____

8. Red-Black
   a. Tell me the definition/properties/requirements for a red-black tree.

   b. In C++, the std::map class (key:value pair storage) is implemented with a red-black tree. When you iterate over its keys, they come out in sorted order. Why is that NOT surprising?

9. AVL trees are just binary search trees with an extra property. What is that property, how is it applied to the tree, and what benefit does it give us.

Name: _____

10. Hashing
    a) What are the required properties of a hash function?

    b) What are the desired properties of a hash function?

    c) Linux doesn't save your passwords. Your sysadm cannot tell you what it is. But the system can still log you in. How does that work? Is it perfectly secure?

    d) Thought experiment: I have a hash function that returns values from 0..999 that I must use. It works fine for my 100 slot table with 20 keys and linear probing (I just do a module 100). But I have a lot of keys and want to use a bigger table. It will clearly work for a table with 1000 slots and 200 inserts. How can I make it work well for a table with 10,000 slots and 2000 inserts?

Name: _____

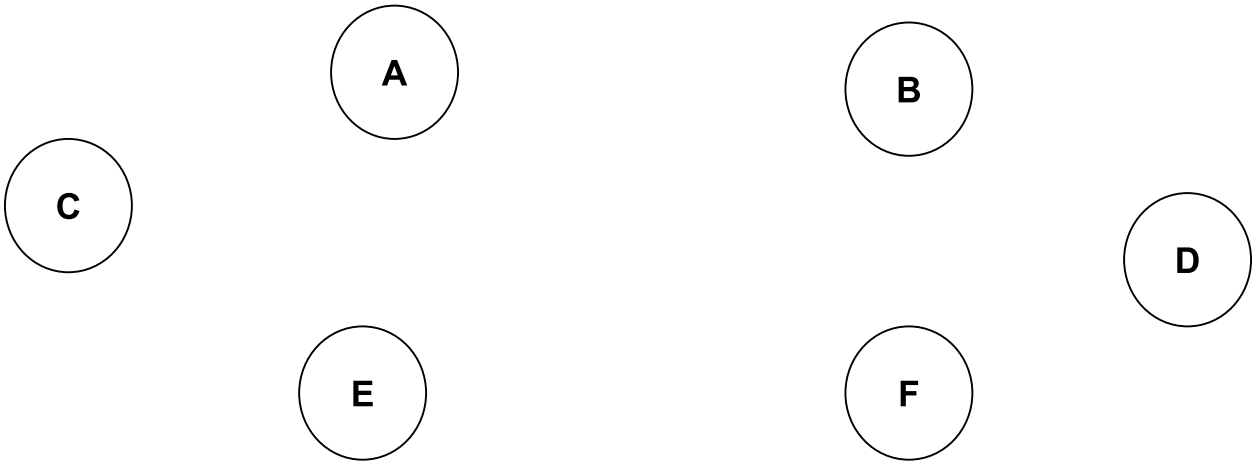
11. My hash function is int $\mathcal{H}$(int x) { return x % 10; } My collision strategy is to apply linear probing. Please insert the keys [86, 95, 55, 105, 17, 116] into this table of size 10.

0	
1	
2	
3	
4	
5	
6	
7	
8	
9	

Name: _____

12. I have a set of nodes in a directed graph represented with an adjacency list. Please draw in the edges. All edges have a weight ***w***=1, so you don't need to include them, just draw the edge.

A: { A, C, F}
B: { D,F,C}
C: {A,D,E,C,F,E}
D: {F}
E: {E}
F: {A,B}



Name: _____

13. Dijkstra's Method
    a. Please list the steps required to perform Djikstra's method on a graph

    b. For the non-directed graph I've given below, please fill in the table using Djkstra's method to find all the shortest paths from start symbol S. I want to see the history of your choices, so cross out prior entries (don't erase). V is just there if you want to track visited nodes - it is not a graded component.

Label	V	Shortest path found so far	Back Path
S			
A			
B			
C			
D			
E			
F			
G