Name: _____

Introduction to Data Structures and Algorithms

Midterm Exam

Instructions: This is s closed-book, closed-notes exam. You have exactly 50 minutes to complete the exam. Indicate your reasoning , including assumptions, in all responses and include all your work. Be as neat as possible. Use the space provided. Attach additional sheets as necessary.

---

1. Determine the stack contents at the points indicated below during the following operations on the stack ADT. Write down the stack contents <u>after</u> the operation on the given line is executed. <u>Be sure to indicate the top of the stack.</u>

```
1     stack<string> s;
2     s.push("a")
3     s.push("z")
4     s.push("zorg")
5     s.push("zero")
6     s.pop()
7     s.push("gift")
8     s.pop()
9     s.pop()
10    s.pop()
11    s.pop()
```

After line 1:   After line 3:

After line 5:   After line 8:

After line 11:

Name: _____

2. Determine the list contents at the points indicated below during the following operations on the OrderedList ADT. Assume the list uses 1-based indexing. Write down the stack contents <u>after</u> the operation on the given line is executed. <u>Be sure to indicate the ordering of the list.</u>

```
1       list<string> myl;
2       myl.insert(1, "a")
3       myl.insert(2, "z")
4       myl.insert(1, "zorg")
5       myl.insert(2, "zero")
6       myl.remove(3)
7       myl.insert(1,"gift")
8       myl.remove(3)
9       myl.remove(1)
10      myl.remove(2)
11      myl.remove(1)
```

After line 1:   After line 3:

After line 5:   After line 8:

After line 11:

3. Consider the following partial C++ code representing a singly-linked list holding integer values.

```cpp
class List
{
    struct Node
    {
      int data;
      Node * next;
    };

    Node *head, *tail;
};
```

a) Write a default constructor for the class List.

3. cont.

b) Write a method with signature `void append(int item)` to append item to the end of the list. Assume all allocations succeed.

4. Write a **recursive** function with the signature

```
void print_permutations(std::string a, std::string b);
```

that when called like

```
print_permutations("", "kayak");
```

would print to standard output all permutations of the string b.

5. Consider the following partial C++ code representing an array-based stack holding integer values.

```
class Stack
{
    int * data;
    int size, top;

public:

    Stack(){
      data = nullptr;
      size = 0;
      top = 0;
    };
};
```

Write a method called void push(int v) to push v onto the stack.