1. Simple Questions.
   1) When we talk about *linked lists*, does it have to do with (choose the most closely related item):
      i. data types (such as priority queues)
      ii. goals (such as adaptability)
      iii. implementation (of, say, a sequence)

   2) Underscore the line(s) (if any) in the following code which potentially break the principle of encapsulation:

```
class Node {
           public Object element;
           public Node next;
           Node() { this(null, null); }
           void setNext(Node newText) { next = newText; }
           Node getNext() { return next; }
}
```

   3) What is the primary reason to use quadratic probing instead of linear probing? (choose the most closely related item):

      i. Easier to expand the hash table when it gets full
      ii. To avoid the primary cluster which may cause excessive data collision
      iii. Achieve the same search time with less memory
      iv. Easier to support delete

4) Suppose that there are N distinct elements in a binary heap (with the maximum at the root). Which positions could possibly be occupied by the fourth largest element?
   i. 1
   ii. 2 or 3
   iii. 4 through 7
   iv. 2 through 15
   v. 16 and higher

5) Characterize, using the big-Oh notation, the worst-case running time of the following algorithm:

Let A be a given array of n integers.

```
for i <- 0 to n-1 do
   for j <- 0 to (i*i)-1 do
     Let A[j mod n] <- j.
   end for
end for
```

Best characterization: O(_______).

2. **(Linked list)** Recall we did singly linked list in PA #4. To refresh your memory,
    - Consider a list with elements (1,2,3,4,5)
        - The list will be like 1→ 2→3→4→5

Singly linked list doesn't have loops. Suppose your friend (call him Bob) has inadvertently introduced a loop into his implementation on PA #4

- Consider a list with elements (1,2,3,4,5)
- Let 3 point to 1 instead of pointing to 4
- The list will be like $$\begin{tikzcd} 1 \arrow[r] & 2 \arrow[r] & 3 \arrow[ll, bend left] & 4 \arrow[r] & 5 \end{tikzcd}$$

Since most of your friends suffer from the same bug while implementing PA#4 you decide to help them by writing a **function** which detects loops in a singly linked list.

- Write program that outputs 'false' if there is no loop in the list input and 'true' if there is a loop in a list output
- Fill in the following code

```
o class list_sorted  {
  public:
    // You need to implement this
    bool FindLoop() {


  }
  private:
    node* pHead;
  };
o struct node{
  public:
  int x;
  node* next;
  };
```

- **Hint**:
    - o Change the node data structure. Have some sort of counter which indicates how many times a node is visited when you traverse through the link list.
    - o There might be better ideas than this!

3. **(Hash table)** You have a hash table of size m = 11 and two has functions h1 and h2:

h1(x) = (sum of the values of the first and last letters of x) mod m
h2(x) = ((value of the last letter) mod (m – 1)) + 1

Where the value of a letter is its position in the alphabet, as in the following table:

a	b	c	d	e	f	g	h	i	g	k	l	m
1	2	3	4	5	6	7	8	9	10	11	12	13
n	o	p	q	r	s	t	u	v	w	x	y	z
14	15	16	17	18	19	20	21	22	23	24	25	26

1) Draw a picture of the resulting hash table after inserting the following words (by the order):
ibex, hare, ape, bat, koala, mud, dog, carp, stork

2) Highlight cells that are looked at when trying to find bird. Do this for each of the following techniques.

Chaining with h1 as your hash function.

Linear probing with h1 as your hash function.

4. **(Heap)** Give an algorithm for changing the value of an arbitrary element from a heap of size N. Also, determine the worst case time complexity of your algorithm. Describe your algorithm in pseudo-code. You may assume that the position of the element whose value to be changed is given.

5. **(Binary tree)** Let T be a binary tree.

1) If T has seven nodes, what is its minimum height?

What is its maximum height?

Draw two trees with seven nodes that achieve the minimum and maximum height, respectively.

2) When deleting a node of a binary search tree, if we decide to replace it with a node in its left subtree (of course, if it has one), which node should we choose?

3) A particular binary search tree has the following known about it:
A pre-order traversal yields 88, 6, 1, 3, 2, 5, 4, 30, 10, 20.
A post-order traversal yields 2, 4, 5, 3, 1, 20, 10, 30, 6, 88.
Draw this tree.

4) What is the in-order traversal of the tree from 3)?

5) Using the node values from 3), draw a minimal-height BST. Make it a complete tree.

6. **(Inheritance)** Consider at the following code [each class is in the appropriately named file]:

```
class Animal {
public:
      virtual void describe() {
            System.out.println("Animal");
      }
}

class Rabbit: public Animal {
public:
      Rabbit() { Animal(); }
      void describe() {
            System.out.println("Rabbit");
      }
      void hop () {
            System.out.println ("Hop");
      }
}

class Dog: public Animal {
public:
      Dog() {}
      void describe() {
            System.out.println("Dog");
      }
}

class Poodle: public Dog {}
```

1) List all constructors called for each line.

Code	**Constructor Calls**
`Animal a = new Dog();`	
`Rabbit r = new Rabbit();`	
`Poodle p = new Poodle();`	
`Animal a2 = a;`	

2) What is output of the following lines? If it is a CT / RT exception, say so. Assume each line is executed independently (no cascading errors).

Code	**Output**
`(new Animal()).describe();`	
`(new Rabbit()).describe();`	
`(new Dog()).describe();`	
`(new Poodle()).describe();`	

`Animal a = new Rabbit();`
`a.describe();`	

`Animal b = new Rabbit();`
`b.hop();`	

7. **(String searching)** This question is about the problem of searching for any occurrence of an M-character pattern in an N-character text string. Suppose that student X uses the brute-force method and student Y uses the right-left scan. The students are each given the option of picking from among one of the three string

generators listed below for both programs to be tested on (both pattern and text to be generated from the same generator, then both programs invoked on the same input).

A. random string of As and Bs
B. random ASCII characters
C. all As except last character is B

Fill in the blanks to give the best choice and expected results from each students' point of view. (If the asymptotic number of character compares is the same, answer "1".)

(a) For X's choice _______, brute-force is a factor of _______ faster than right-left.
(b) For Y's choice _______, right-left is a factor of _______ faster than brute-force.

8. **(Operator overloading)** Provide the declarations of the overloaded operator+ for F1 that will permit the two add operations shown in main().

```
class F1 {
public:
   int data;







};


main(){
 F1 a;
 a=a+1;
 a=1+a;
}
```

9. Indicate true or false.

```
class A1 {
 private:
   int a;
   char* name;
 public:
   A1(char* n){name=new char[strlen(n)+1];
   memcpy(name, n, strlen(n)+1);}
   int b;};
class D1 : public A1{
 public:
   int data;
};
class D2 : private A1{};
```

a. The “public” constructor for A1 allocates memory from global heap for the private member “name” therefore “name” can be direcly accessed by class D2, which is derived from A1.

b. Since D2 inherits A1 as a public class, both the private members of A1 are directly accessible within D2.

10. **(set)** Suppose the *set s* is declared with

```
set<int> s;
```

Write the definition of the function *removeOdd*() that will remove all odd integers belonging to *s* with the call

```
removeOdd(s);
```

11. **(doubly linked list**) (a) Why do some programmers like to have dummy nodes at each end of a doubly linked list? (b) What are the disadvantages of having dummy nodes at the beginning of the list?

12 **(map)** Suppose we want a data structure that allows us to look up a person's name and address from his or her social security number. (a) How would you use an *STL map* for this? (b) How long does it take to access the name and/or address from a social security number in this data structure?