Exam 2 - Solution

Part I - Circle the best answer for the following questions (50%)

1. Linear search can be applied to
a. only sorted lists b. any list of integers **c. any list of objects that are comparable**
d. none of the above
2. which one of the following conditions cause linear search to perform worst?
a. the target is at the end of the list b. target is not in the list **c. all of the above**
3. The complexity of linear search is
a. Order(n) b. order(log n) c. order(n log n)
4. When applying binary search, after each comparison, the size of the list is reduced by
a. one entry **b. half** c. third d. cannot determine, it depends on the data set
5. Many algorithms perform efficiently is data is already sorted
a. TRUE b. FALSE
6. Fast sorts are of order
a. n b. log n **c. n log n** d. n^2
7. Bubble sort works well for small data sets
a. TRUE b. FALSE
8. Selection sort may perform better than bubble sort, because
a. number of swaps is less b. number of comparisons is less c. no difference in swaps or comparisons
9. Consider the following code used in quick sort

```
void Foo(int[] A, int left, int right) {
        if (A[left] > A[right])
                Swap(A[left], A[right]);
}
```

the purpose of the code is to
a. pivot b. partition c. find the maximum
10. Which one of the following best describes the recurrence relation in quicksort?
a. C(n) = 2C(n/2) + n b. C(n) = n c. C(n) = 3C(n/2)
11. The JAVA IO statement **File f1 = new File("myData.txt");**
a. creates a new file mydata.txt **b. creates a reference to a file object**
12. The predefined **BufferedInputStream** object to represent a stream of input that comes from the keyboard is called
a. System.out **b. System.in** c. System.out.println()
13. The line : **URL u = new URL("http://www.whitehouse.gov")**
a. Sets up communication software on your machine b.Contacts the remote machine
c. Waits for response d. Sets up connection **e. All of the above**
14. Which one of the following is not a primitive data type
a. int **b. class** c. char d.boolean

15. The main purpose of inheritance is
a. defining new objects **b. deriving new classes from existing ones** c. defining protected members

16. A difference between private and protected member of a class makes sense when we are dealing with
a. Defining new members of a class b. **using inheritance to create new classes**
c. when we want to protect private members from changes
17. A key idea of inheritance is
a. recursion **b. software reuse** c. defining a "has-a" relationship

18. In the declaration : *class Circle extends Shape* the base class is
a. Circle **b. Shape** c. both Shape and Circle

19. Which one of the members are inherited?
a. public only b. protected only **c. public and protected** d. public, private and protected

20. Constructors in base class are inherited by derived class

a. TRUE b. **FALSE**

Part II - WRITE NEW METHODS (30%)

1. (15 points) Study the class definition of Set

```
public class Set {
   private Vector list;
   public Set() { ….}
   public void add(Object obj);
   public boolean contains(Object obj){..}
   public Set union(Set S){…}
   public Set intersection(Set S){…}
   public Set complement(Set S){…}
   public void print();
   boolean isEmpty();
   ….
}
```

a. Complete the method **contains (hint: use vector methods)**

```
public boolean contains(Object obj){
      return (list.contains(obj));
}
```

b. Complete the method isEmpty( ) {

```
      return (list.size() == 0);
   }
```

c. Suppose we are dealing with a set of integers. eg: S = {-2, -1, 0, 1, 2}

A set is closed with respect to addition if the sum of any two elements is also another element in the set. Note the above set is closed under addition. Write a method, that returns true if the set is closed under addition. You may assume any of the above methods exists.

```
public boolean  isClosedUnderAddition( ) {
      for (int I=0; I<list.size() ; I++)
        for (int j=0; j<list.size() ; j++)
           {    int  n1 =  ((Integer)list.elementAt(i)).intValue();
                int  n2 =  ((Integer)list.elementAt(j)).intValue();
                Integer Sum = new Integer(n1+n2);
                if  (!list.contains(Sum))  return false;
           }
       return true;
}
```

2. (15 points) Consider the following class definition

```
? public class Account {
      protected double balance;
      protected  long accountNumber;
      public Account(){}
      public Account(double b, long n){}
      public void deposit(double b){}
      public void withdraw(double b){}
      public void getBalance(double b){}
  };
```

a. Complete the methods deposit.

```
public void deposit(double b){
      if (b>0)   balance += b;
}
```

b. Complete the method withdraw (make sure you cannot withdraw money you don't have)

```
 public void withdraw(double b){
      if  (b <= balance)
          deposit -= b;
  }
```

c. Derive a new class SavingsAccount with new members, instance variable interestRate, and a public method calculateInterest( ) . Complete the method calculateInterest( ) - interest = balance * monthlyrate

```
class SavingsAccount extends Account {
         double interestRate;
         public double calculateInterest() {
               int interest;
               interest = balance*interestRate/12;
               balance += interest;
               return interest;
}
```

Part III - TRACE THE CODE (20%)

1. (10 points) consider the following code. A is an array of integers.

```
public int foo(int target){
   for (int I=0;I<A.size();I++)
     if (A.elementAt(I)==target)
              return i;
   return -1;
}
```

a. What is the purpose of the method foo?
   This is the linear search algorithm

b. If the array A is symmetric, i..e left and right halves are the same. egs: [1 2 3 2 1] or [2 3 3 2]   Modify the code above to achieve the same goals.

```
    for (int I=0;I<A.size()/2;I++)
```

2. (10 pts) Consider the following code

```
public int binarySearch(Object target){
   int first=0, last=A.size()-1;
   middle=(first+last)/2;
   while (first<=last){
     if (A.elementAt(middle) > target)
          {last=middle-1;}
     else if (A.elementAt(middle) < target)
          {first=middle+1;}
     else return middle;
   }
   return -1
}
```

a. What is wrong with the code? (hint: one line misplaced)

middle=(first+last)/2;
should be inside the while loop

b. Consider the array A = {1, 4, 5, 6, 8, 10, 12}, if the target is 4, show the values of middle entry as you run through the above code. How many comparisons are needed to decide that 4 is a part of the array.

round	**first**	**last**	**middle index**	**middle entry**	**search**
1	**0**	**7**	**3**	**6**	**left**
2	**0**	**2**	**1**	**4**	**complete (target found)**