University Interscholastic League

Computer Science Competition

Number 105 (Regional - 2007)

General Directions (Please read carefully!):

1) DO NOT OPEN EXAM UNTIL TOLD TO DO SO.

2) **NO CALCULATORS OF ANY KIND MAY BE USED.**

3) There are 40 questions on this contest exam. You have 45 minutes to complete this contest. If you are in the process of actually writing an answer when the signal to stop is given, you may finish writing that answer.

4) Papers may not be turned in until 45 minutes have elapsed. If you finish the test before the end of the allotted time, remain at your seat and retain your paper until told to do otherwise. You may use this time to check your answers.

5) All answers must be written on the answer sheet/Scantron card provided. Indicate your answers in the appropriate blanks provided on the answer sheet or on the Scantron card. Clean erasures are necessary for accurate Scantron grading.

6) You may place as many notations as you desire anywhere on the test paper, but not on the answer sheet or Scantron card which are reserved for answers only.

7) You may use additional scratch paper provided by the contest director.

8) All questions have ONE and only ONE correct (BEST) answer. There is a penalty for all incorrect answers. **All provided code segments are intended to be syntactically correct, unless otherwise stated. Ignore any typographical errors and assume any undefined variables are defined as used.**

9) A reference to commonly used Java classes is provided at the end of the test, and you may use this reference sheet during the contest. You may detach the reference sheets from the test booklet, but DO NOT DO SO UNTIL THE CONTEST BEGINS.

10) Assume that any necessary import statements for standard Java 2 packages and classes (e.g. .util, System, Math, Double, etc.) are included in any programs or code segments that refer to methods from these classes and packages.

Scoring:

1) All questions will receive **6 points** if answered correctly; no points will be given or subtracted if unanswered; **2 points** will be deducted for an incorrect answer.

QUESTION 1

What is the sum of $\texttt{BC}_{16}$ and $\texttt{15}_{16}$?

A. $\texttt{11010001}_2$ B. $\texttt{11110001}_2$ C. $\texttt{11010010}_2$ D. $\texttt{11000010}_2$ E. $\texttt{11010000}_2$

QUESTION 2

What is output by the code to the right?

A. `3` B. `13` C. `6`

D. `3.33333` E. `16`

```
int x = 10;
int y = 3;
int z = x % 13 + x / y;
System.out.print( z );
```

QUESTION 3

What is output by the code to the right?

A. `1200` B. `600` C. `420`

D. `630` E. `10886400`

```
int n = 20;
int total = 0;
for(int i = 1; i <= n; i++)
    for(int j = 1; j <= i; j++)
        for(int k = 0; k < 3; k++)
            total++;
System.out.print( total );
```

QUESTION 4

What is output by the code to the right?

A. `10isa6` B. `kisa6`

C. `10isa33` D. `kisaf`

E. There is no output due to a syntax error in the code.

```
int x = 3;
int y = 13;
String s1 = "isa";
String s2 = y - x + s1 + x + x;
System.out.println( s2 );
```

QUESTION 5

What is output by the code to the right?

A. `012345`

B. `542342`

C. `542241`

D. `142245`

E. `542151`

```
int size = 6;
int[] st1 = {9, 16, 0, 1, 25, 4};
int[] st2 = new int[size];
for(int i = 0; i < size; i++)
    st2[i] = (int)Math.sqrt( st1[i] );
int[] st3 = new int[size];
for(int i = 0; i < size; i++){
    st3[ st2[i] ]++;
    st3[ st1[i] % size ]++;
    st3[ st1[i] / size ]++;
}
for(int i : st3 )
    System.out.print(i);
```

QUESTION 6

What is output by the code to the right?

A. `larskimejakeashe`

B. `sralemikekajehsa`

C. `lkjaaiasrmkhseee`

D. `ehsaekajemiksral`

E. `seeermkhaiaslkja`

```
int lim = 4;
char[][] tab = new char[lim][lim];
String[] nms = {"lars", "kime", "jake",
                                     "ashe"};
for(int i = lim - 1; i >= 0; i--)
  for(int j = 0; j < lim; j++)
    tab[i][j] = nms[j].charAt(lim - i - 1);

for(char[] row : tab)
  for( char c : row )
    System.out.print(c);
```

QUESTION 7

What is output by the code to the right?

A. `29`

B. `32`

C. `27`

D. There is no output due to a syntax error.

E. There is no output due to a runtime error.

```
int[] vs = {-3,-2,4,5,7,-2,-1,3,-5,-4,2};
int tot = 0;
int i = 0;
while( i < vs.length ){
  if( (vs[i] < -1) || (vs[i] * vs[i]++ < 20))
    tot += Math.abs( vs[i] );
  else
    tot += 2;
  i++;
}
System.out.print( tot );
```

QUESTION 8

Assume `x`, `y`, and `z` are `int` variables. Which answer is logically equivalent to this Boolean expression?

```
!( x + y < z && x * y >= z )
```

A. `!(x + y < z) && !( x * y >= z)`

B. `x + y < z || x * y >= z`

C. `x + y >= z || !(x * y >= z)`

D. `!( x + y >= z) || !(x * y < z)`

E. More than one of these.

QUESTION 9

What is output by the code to the right?

A. `0.77_100` B. `77_100`

C. `0_23` D. `0_100`

E. There is no output due to a syntax error.

```
int x2 = 100;
int y2 = 77;
int z2 = y2 / x2;
System.out.print( z2 + "_" + x2 );
```

QUESTION 10

What is output by the method call `ht(17, 31)` ?

A. `31_17` B. `0_0`

C. `17_31` D. `x_y`

E. The output cannot be predicted due to overflow of the variables.

```
public static void ht(int x, int y){
  x = x ^ y;
  y = y ^ x;
  x = y ^ x;
  System.out.print( x + "_" + y );
}
```

QUESTION 11

Which of the following statements will not cause a syntax error?

I. `Vehicle v = new Vehicle();`

II. `Bike b1 = new Vehicle();`

III. `Bike b2 = new MountainBike(14);`

A. I only B. II only C. III only

D. I and III E. II and III

QUESTION 12

What is output by the following code?

```
MountainBike m1 = new MountainBike(10);
int w = m1.wheels();
System.out.print( m1.grs() + "_" + w );
```

A. `myGears_2`

B. `2_10`

C. `myGears_w`

D. `2_0`

E. `10_2`

QUESTION 13

What is output by the following code?

```
MountainBike m2 = new MountainBike(10);
m2.show();
```

A. `mountain` B. `m2` C. `engine:`

D. `10` E. `vehicle`

QUESTION 14

What is output by the following code?

```
Vehicle v1 = new MountainBike(10);
System.out.print( v1 );
int val = ((MountainBike)v1).grs();
System.out.print( val );
```

A. `engine:false10`

B. `engine:`

C. `engine:10`

D. There is no output due to a syntax error.

E. There is no output due to a runtime error.

```
public interface Bike{
  public int wheels();
  public String toString();
}

------------------------------------------

public abstract class Vehicle{

  public abstract boolean humanPower();

  public void show(){
    System.out.print( type() );
  }

  private String type(){
    return "vehicle";
  }

  public String toString(){
    return "engine:" + !humanPower();
  }
}

------------------------------------------

public class MountainBike extends Vehicle
                            implements Bike{

  private int myGears;

  public MountainBike(int gears){
    myGears = gears;
  }

  public boolean humanPower(){
    return true;
  }

  public int wheels(){
    return 2;
  }

  public String type(){
    return "mountain";
  }

  public int grs(){
    return myGears;
  }
}
```

QUESTION 15

What is output by the code to the right?

A. `4.5`

B. `5.0`

C. `5`

D. `5.75`

E. `5.25`

```
int div = 2;
double a = 5 / div + 1.5 + 7 / (div * 2);
System.out.println( a );
```

QUESTION 16

What is output by the method call `change(11)` ?

A. `11` B. `100` C. `1`

D. `102` E. `201`

QUESTION 17

Which of the following best describes what will occur if the precondition of method `change` is not met?

A. An `IllegalArgumentException` will be thrown.

B. The method call will result in an infinite loop.

C. A stack overflow will eventually occur.

D. A syntax error will occur.

E. The value of the parameter `n` will be printed out.

QUESTION 18

Which of the following best describes what method `change` does if the precondition is met?

A. It prints out the value of `n` in base 3.

B. It prints out the value of `n` in base 3, but with the digits reversed.

C. It prints out `n` `1`'s if `n` is prime.

D. It prints out all the factors of `n`.

E. It prints out the first 3 multiples of `n`.

```
//pre: n >= 0
public static void change(int n){
  if( n <= 2 )
    System.out.print( n );
  else{
    change( n / 3 );
    System.out.print( n % 3 );
  }
}
```

QUESTION 19

What is output by the code to the right?

A. `1315,6,6,7000,`

B. `1315,6,6,2731,7000,`

C. `1315,6,6,7,3,`

D. `1315,6,6,2731,7,3,`

E. `1315,,6,,6,,7000,`

```
String input = "1315..6..6.aab.7e3";
String[] res = input.split("\\D+");
for(String s : res)
  System.out.print( s + "," );
```

QUESTION 20

What sorting algorithm is implemented by the static methods to the right?

A. Quick sort

B. Selection sort

C. Merge sort

D. Insertion sort

E. Heap sort

```
public static void sort(int[] data) {
  int[] temp = new int[data.length];
  sort(data, temp, 0, data.length - 1);
}

public static void sort(int[] data,
                    int[] temp, int i, int j){

  if( i < j){
    int m = (i + j) / 2;
    sort(data, temp, i, m);
    sort(data, temp, m + 1, j);

    int le = m;
    int tp = i;
    int ne = j - i + 1;
    while( i <= le && m + 1 <= j){
      if( data[i] <= data[m + 1] ){
        temp[ tp ] = data[ i ];
        i++;
      }
      else{
        temp[tp] = data[m + 1];
        m++;
      }
      tp++;
    }

    while( i <= le){
      temp[ tp ] = data[ i ];
      tp++;
      i++;
    }

    while( m + 1 <= j){
      temp[ tp ] = data[ m + 1 ];
      tp++;
      m++;
    }

    for(int k = 0; k < ne; k++){
      data[ j ] = temp[ j ];
      j--;
    }
  }
}
```

QUESTION 21

A sort is defined to be *stable* if equal elements in the original array maintain their relative positions in the sorted array. For example consider the following array of `int`s.

`{0, 7, 5, 3, 7}`

A stable sort ensures that in the sorted array, the 7 originally at index 1 will always be before the 7 originally at index 4. When is the sort implemented to the right stable?

A. Never.

B. Always.

C. Only if the data is already sorted in ascending order.

D. Only if the data is already sorted in descending order.

E. It is not possible to determine if the sorting algorithm is stable or not.

QUESTION 22

It takes method `sort` 10 seconds to sort an array of 1,000,000 unique elements in random order on a given computer. What is the expected time for method `sort` to sort an array of 2,000,000 unique elements in random order on the same computer?

A. 5 seconds B. 21 seconds C. 40 seconds

D. 60 seconds E. 80 seconds

QUESTION 23

What is the running time of method `max` for a `LinkedList` containing `N` items? Choose the most restrictive correct answer.

A. O(1) B. O(N) C. O(NlogN)

D. $O(N^2)$ E. $O(N^3)$

```
//pre: data.size() > 0
public int max(LinkedList<Integer> data){
  int result = data.getFirst();
  for(int i = 1; i < data.size(); i++){
    int val = data.get(i);
    if( val > result )
      result = val;
  }
  return result;
}
```

QUESTION 24

Consider the `Node` and `Structure` classes to the right. What is output by the following code?

```
Structure t = new Structure();
String data = "BALLOON";
for(int i = 0; i < data.length(); i++)
  t.add( data.substring(i, i+1) );
t.show();
```

A. `ANOLB` B. `BALON` C. `ABLON`

D. `ANOOLLB` E. `ABLLNOO`

QUESTION 25

What type of data structure does the `Structure` class implement?

A. A binary search tree.

B. A linked list.

C. A min heap.

D. A max heap.

E. A hash table.

QUESTION 26

What is output by the following code?

```
Structure s = new Structure();
String data2 = "DELTABIG";
for(int i = 0; i < data2.length(); i++)
  s.add( data2.substring(i, i+1) );
System.out.print( s.ct() );
```

A. `0` B. `1` C. `2`

D. `3` E. `4`

QUESTION 27

Which of the following best describes what method `ct` in class `Structure` returns?

A. The number of `Nodes` in the `Structure`.

B. The number of `left` and `right` references in the `Structure` that are equal to `null`.

C. The number of `Nodes` in the `Structure` whose `left` and `right` references are both not `null`.

D. Method `ct` always returns `0`.

E. The number of `Nodes` in the `Structure` whose `left` and `right` references are both `null`.

```
public class Node{
  public String val;
  public Node ft;
  public Node rt;
}
------------------------------------------
public class Structure{

  private Node n;

  public void add(String s){
    n = add(s, n);
  }

  private Node add(String s, Node n){
    if( n == null ){
      n = new Node();
      n.val = s;
    }
    int c = n.val.compareTo( s );
    if( c < 0 )
      n.rt = add(s, n.rt);
    else if( c > 0 )
      n.ft = add( s, n.ft);
    return n;
  }

  public void show(){
    show(n);
  }

  private void show(Node n){
    if( n != null ){
      show(n.ft);
      show(n.rt);
      System.out.print( n.val );
    }
  }

  public int ct(){
    return ct(n);
  }

  private int ct(Node n){
    int res = 0;
    if( n != null ) {
      if( n.ft == null && n.rt == null )
        res = 1;
      else
        res = ct(n.ft) + ct(n.rt);
    }
    return res;
  }

}
```

QUESTION 28

What replaces `<*1>` in the code to the right to set the `Reptile`'s object `name` field to the parameter `nm`?

A. `super(nm)`

B. `this(nm)`

C. `Animal(nm)`

D. `super.name = nm`

E. More than one of these.

```
public class Animal{

  private String name;

  public Animal(){
    name = "unknown";
  }

  public Animal(String nm){
    name = nm;
  }

  public boolean equals(Object other){
    return other instanceof Animal &&
       name.equals( ((Animal)other).name );
  }
}
-----------------------------------------
public class Reptile extends Animal{

  private boolean swims;

  public Reptile(String nm, boolean sms){
    <*1>;
    swims = sms;
  }

}
```

Assume `<*1>` is filled in correctly.

QUESTION 29

What is output by the following code?

```
Animal a = new Animal("max");
Reptile r = new Reptile("max", true);
System.out.print( a.equals(r) );
System.out.print( r.equals(a) );
```

A. `falsefalse`

B. `falsetrue`

C. `truefalse`

D. `truetrue`

E. `true` and then a runtime error occurs,

QUESTION 30

What is output by the code to the right?

A. `11131517`

B. `17151311`

C. `1115`

D. `91115`

E. `1317`

```
ArrayList<Integer> list = new
                      ArrayList<Integer>();
int inc = 2;
for(int j = 5; j >= 0; j--){
  list.add( j + inc );
  inc += 3;
}
Iterator<Integer> it = list.iterator();
while( it.hasNext() ){
  if( it.next() > 10 )
    System.out.print( it.next() );
}
```

QUESTION 31

What is output by the code to the right?

A. `141720233`

B. `3312`

C. `332072411`

D. `332012147`

E. `3317`

```
ArrayList<Integer> kd = new
                      ArrayList<Integer>();
int[] data = {1, 4, 1, 7, 20, 2, 33};
for(int el : data){
  if( el < kd.size() )
    kd.add(el, el);
  else if( el < 20 )
    kd.add(el);
  else
    kd.add(0, el);
}
for(int el : kd)
    System.out.print( el );
```

QUESTION 32

What is output by the following code?

```
int[] dt = {0, 1, 2, 3, 4, 10};
System.out.print( myst(dt, 1) );
```

A. `0`   B. `1`

C. `-1`   D. `10`

E. There is no output due to a runtime error.

QUESTION 33

What is output by the following code?

```
int[] dt = {1, 0, 3, 4, 2};
System.out.print( myst(dt, 3) );
```

A. `-1`   B. `0`

C. `1`   D. `2`

E. There is no output due to a runtime error.

QUESTION 34

Which of the following best describes the function of method `myst`?

A. Count the number of elements in `lt` equal to `tgt`.

B. Sort the array `lt`.

C. Search the array `lt` for `tgt`.

D. Find the maximum value in the array `lt`.

E. Find the minimum value in the array `lt`.

```
/* pre: list != null, list.length > 0,
   list is sorted in ascending order.
*/
public static int myst(int[] lt, int tgt){

  int len = lt.length;
  if( tgt < lt[0] || tgt > lt[len - 1] )
    return -1;

  int pos = tgt - lt[0];
  pos *= len - 1;
  pos /= lt[len - 1] - lt[0];
  int inc = lt[pos] < tgt ? 1 : -1;

  while( pos >= 0 && pos < len &&
                          lt[pos] != tgt){

      pos += inc;
  }

  pos = (pos == len) ? -1 : pos;
  return pos;
}
```

QUESTION 35

What is output by the code to the right?

A. `254434`

B. `25443`

C. `ABDOR`

D. `34524`

E. There is no output due to a runtime error.

```
char c;
Map<Character, Integer> m =
          new TreeMap<Character,Integer>();

String names = "ROB_BOB_BRAD_DAD_BROOD";

for(int i = 0; i < names.length(); i++){
  c = names.charAt(i);
  if( Character.isLetter(c) ){
    if( m.containsKey(c) )
      m.put( c, m.get(c) + 1 );
    else
      m.put( c, 1 );
  }
}

Set<Character> st = m.keySet();
for( Character k : st )
  System.out.print( m.get(k) );
```

QUESTION 36

What replaces **<*1>** in the code to the right so that `myCon` only stores `ArrayList`s of `String`s?

A. `<ArrayList<String>>`

B. `<String>`

C. `<ArrayList::String>`

D. `<String[]>`

E. `<ArrayList>`

Assume **<*1>** is filled in correctly.

QUESTION 37

What is output by the following code?

```
Structure2 str = new Structure2(5);
str.add("A", 3);
str.add("B", 1);
str.add("C", 1);
str.add("D", 2);
while( !str.isEmpty() )
  System.out.print( str.remove() );
```

A. ABCD B. DCBA

C. BCDA D. CBDA

E. There is no output due to a runtime error.

QUESTION 38

What is output by the following code?

```
Structure2 str2 = new Structure2(5);
str2.add("A", 10);
str2.add("B", 1);
while( !str2.isEmpty() )
  System.out.print( str2.remove() );
```

A. AB B. BA

C. AA D. BB

E. There is no output due to a runtime error.

QUESTION 39

What type of data structure does the `Structure2` class implement?

A. A list. B. A priority queue.

C. A stack. D. A hash table.

E. A binary search tree.

```
public class Structure2{

  private ArrayList<*1> myCon;

  //pre: h > 0
  public Structure2(int h){
    myCon = new ArrayList<*1>();
    for(int i = 0; i <= h; i++)
      myCon.add( new ArrayList<String>() );
  }

  //pre: 0 <= val <= high()
  public void add(String s, int val){
    myCon.get(val).add(s);
  }

  public boolean isEmpty(){
    boolean e = true;
    int i = 0;
    while( e && i <= high() ){
      e = myCon.get(i).size() == 0;
      i++;
    }
    return e;
  }

  // pre: !isEmpty()
  public String remove(){
    return helper(0);
  }

  // pre: !isEmpty()
  public String get(){
    return helper(1);
  }

  private String helper(int op){
    int i = 0;
    boolean done = false;
    String result = "";
    while( !done && i <= high() ){
      if( myCon.get(i).size() != 0 ){
        done = true;
        if( op == 0 )
          result = myCon.get(i).remove(0);
        else
          result = myCon.get(i).get(0);
      }
      else
        i++;
     }
     return result;
  }

  public int high(){
    return myCon.size() - 1;
  }
}
```

QUESTION 40

What is output by the following code?

```
Sale s1 = new Sale(5);
Sale s2;
Sale s3 = new Sale();
Sale s4 = new Sale(5, "book");
s2 = s4;
s4 = new Sale(1, "food");
System.out.println( Sale.getCount() );
```

A. `0`

B. `4`

C. `7`

D. `10`

E. `11`

```
public class Sale{

  private static int count = 0;
  private int amount;
  private String item;

  public Sale(){
    this(0);
    count++;
  }

  public Sale(int amt){
    this(amt, "unknown");
    count++;
  }

  public Sale(int amt, String it){
    amount = amt;
    item = it;
    count++;
  }

  public static int getCount(){
    return count;
  }

}
```

No Material on this page.

Standard Classes and Interfaces — Supplemental Reference

`class java.lang.Object`

- `boolean equals(Object other)`
- `String toString()`
- `int hashCode()`

`interface java.lang.Comparable`

- `int compareTo(T other)`
  // return value < 0 if `this` is less than `other`
  // return value = 0 if `this` is equal to `other`
  // return value > 0 if `this` is greater than `other`

`class java.lang.Integer implements java.lang.Comparable`

- `Integer(int value)`
  // constructor
- `int intValue()`
- `boolean equals(Object other)`
- `String toString()`
- `int compareTo(Integer anotherInteger)`
  // specified by `java.lang.Comparable`
- `static int parseInt(String s)`
  // Parses the string argument as a signed decimal integer.

`class java.lang.Double implements java.lang.Comparable`

- `Double(double value)`
  // constructor
- `double doubleValue()`
- `boolean equals(Object other)`
- `String toString()`
- `int compareTo(Double anotherDouble)`
  // specified by `java.lang.Comparable`

`class java.lang.String implements java.lang.Comparable`

- `int compareTo(String anotherString)`
  // specified by `java.lang.Comparable`
- `boolean equals(Object other)`
- `int length()`
- `String substring(int from, int to)`
  // returns the substring beginning at `from`
  // and ending at `to-1`
- `String substring(int from)`
  //returns `substring(from, length())`
- `int indexOf(String s)`
  // returns the index of the first occurrence of `s;`
  // returns `-1` if not found
- `int indexOf(String str, int fromindex)`
  // Returns the index within this string of the first occurrence
  // of the specified substring, starting the
  // search at the specified index.`char`
- `charAt(int index)`
  // Returns the character at the specified index.
- `int indexOf(int ch)`
  // Returns the index within this string of the first occurrence
  // of the specified character.
- `int indexOf(int ch, int fromindex)`
  // Returns the index within this string of the first occurrence
  // of the specified character, starting the
  // search at the specified index.
- `String toLowerCase()`
  // Converts all of the characters in this `String` to lower
  // case using the rules of the default locale.
- `String toUpperCase()`
  // Converts all of the characters in this `String` to upper
  // case using the rules of the default locale.
- `String[] split(String regex)`
  // Splits this string around matches of the given regular
  // expression.

`class java.lang.Character`

- `static boolean isDigit(char ch)`
- `static boolean isLetter(char ch)`
- `static boolean isLetterOrDigit(char ch)`
- `static boolean isLowerCase(char ch)`
- `static boolean isUpperCase(char ch)`
- `static char toUpperCase(char ch)`
- `static char toLowerCase(char ch)`

`class java.lang.Math`

- `static int abs(int x)`
- `static double abs(double x)`
- `static double pow(double base, double exponent)`
- `static double sqrt(double x)`
- `static double ceil(double a)`
- `static double floor(double a)`
- `static double min(double a, double b)`
- `static double max(double a, double b)`
- `static int min(int a, in b)`
- `static int max(int a, int b)`
- `static long round(double a)`

`class java.util.Random`

- `int nextInt()`
- `double nextDouble()`

`interface java.util.List<E>`

- `boolean add(E x)`
- `int size()`
- `Iterator<E> iterator()`
- ListIterator<E> listIterator()

`class java.util.ArrayList<E> implements java.util.List<E>`

`interface java.util.Iterator<E>`

- Methods in addition to the `List` methods:
- `E get(int index)`
- `E set(int index, E x)`
  // replaces the element at `index` with `x`
- `void add(int index, E x)`
  // inserts `x` at position `index`, sliding elements
  // at position `index` and higher to the right
  // (adds 1 to their indices) and adjusts `size`
- `E remove(int index)`
  // removes element from position `index`, sliding elements
  // at position `index + 1` and higher to the left
  // (subtracts 1 from their indices) and adjusts `size`

`class java.util.LinkedList<E> implements java.util.List<E>`

- Methods in addition to the `List` methods
- `void addFirst(E x)`
- `void addLast(E x)`
- `E getFirst()`
- `E getLast()`
- `E removeFirst()`
- `E removeLast()`

`interface java.util.Set<E>`

- `boolean add(E x)`
- `boolean contains(Object x)`
- `boolean remove(Object x)`
- `int size()`
- `Iterator<E> iterator()`

`class java.util.HashSet<E> implements java.util.Set<E>`

`class java.util.TreeSet<E> implements java.util.Set<E>`

`interface java.util.Map<K,V>`

- `Object put(K key, V value)`
- `V get(Object key)`
- `boolean containsKey(Object key)`
- `int size()`
- `Set<K> keySet()`
- `Set<Map.Entry<K, V>> entrySet()`

`class java.util.HashMap<K,V> implements java.util.Map<K,V>`

`class java.util.TreeMap<K,V> implements java.util.Map<K,V>`

`interface java.util.Map.Entry<K,V>`

- `K getKey()`
- `V getValue()`
- `V setValue(V value)`

- `boolean hasNext()`
- `E next()`
- `void remove()`

`interface java.util.ListIterator<E> extends java.util.Iterator<E>`

- Methods in addition to the `Iterator` methods
- `void add(E x)`
- `void set(E x)`

`class java.lang.StringBuffer`

- `StringBuffer append(char c)`
- `StringBuffer append(String str)`
- `StringBuffer append(StringBuffer sb)`
- `int capacity()`
- `char charAt(int index)`
- `StringBuffer delete(int start, int end)`
- `StringBuffer deleteCharAt(int index)`
- `StringBuffer insert(int offset, char c)`
- `StringBuffer insert(int offset, String S)`
- `int length()`
- `void setCharAt(int index, char ch)`
- `String substring(int start)`
- `String substring(int start, int end)`
- `String toString()`

`class java.lang.Exception`

- `Exception()`
- `Exception(String message)`

`class java.util.Scanner`

- `Scanner(InputStream source)`
- `boolean hasNext()`
- `boolean hasNextInt()`
- `boolean hasNextDouble()`
- `String next()`
- `int nextInt()`
- `double nextDouble()`
- `String nextLine()`
- `Scanner useDelimiter(String pattern)`
  `//` Sets this scanner's delimiting pattern to a pattern
  `// constructed from the specified`
  `// String.`

Computer Science Answer Key
UIL Invitational Regional 2007

1.	A	11.	C	21.	B	31.	D
2.	B	12.	E	22.	B	32.	B
3.	D	13.	E	23.	D	33.	A
4.	C	14.	A	24.	A	34.	C
5.	C	15.	A	25.	A	35.	B
6.	E	16.	D	26.	D	36.	A
7.	B	17.	E	27.	E	37.	C
8.	C	18.	A	28.	A	38.	E
9.	D	19.	C	29.	D	39.	B
10.	A	20.	C	30.	E	40.	C

Notes:

13. E. The private method `type` in the `Vehicle` class shadows the public method also named `type` in the `MountainBike` class.

22. B The merge sort is O(NlogN) so when the number of elements being sorted doubles, it will take a little more than double the time to do the sorting.

23. D The `LinkedList get` method is average case O(N). N calls to an O(N) method results in $O(N^2)$.

The clause "Choose the most restrictive correct answer." is necessary because per the formal definition of Big O, an algorithm that is $O(N^2)$ is also $O(N^3)$, $O(N^4)$, and so forth.