University Interscholastic League

Computer Science Competition

Number 122 (District 2 - 2010)

General Directions (Please read carefully!):

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

2) **NO CALCULATOR 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. 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 packages and classes (e.g. `.util`, `ArrayList`, 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 671_8 and 111_8?

A. 1002_8 B. 700_8 C. 702_8 D. 772_8 E. 1112_8

QUESTION 2

What is output by the code to the right?

A. 15 B. 27 C. 9

D. 3 E. 12

```
int x = 3;
x += x * 2;
System.out.print(x);
```

QUESTION 3

What is output by the code to the right?

A. 10 B. 11 C. 12

D. 0 E. 13

```
int ticks = 0;
for(int j = 1; j < 12; j++)
  ticks++;
System.out.print(ticks);
```

QUESTION 4

What is output by the code to the right?

A. 15 B. 5 C. 7

D. 6 E. 8

```
String tux = "Linus--Torvalds--";
System.out.print( tux.indexOf("--") );
```

QUESTION 5

What is output by the code to the right?

A. 5 B. 0 C. 1

D. 3 E. 33

```
int[] lows = {-1, -4, -1, 4, 32};
lows[3]++;
System.out.print(lows[3]);
```

QUESTION 6

What is output by the code to the right?

A. 3 B. 2 C. 6.5

D. 6 E. 8

```
int w = 3;
int z = 2;
System.out.print( z + w / z % (w * z) );
```

QUESTION 7

What is output by the code to the right?

A. false false

B. false true

C. true false

D. true true

E. false true true false

```
boolean p = false;
boolean q = true;
boolean r = p || q;
System.out.print((p && q) + " ");
System.out.print(q && !r);
```

QUESTION 8

What is output by the code to the right?

A. 13 B. 14 C. 23

D. 24 E. 1234

```
int x = 4;
int y = 8;
if( x > y || x == y)
  System.out.print(1);
else
  System.out.print(2);
if( x * y > x + y )
  System.out.print(3);
else
  System.out.print(4);
```

QUESTION 9

How many constructors does the `StudentTicket` class have?

A. 0 B. 1 C. 2

D. 3 E. 4

QUESTION 10

What is output by the client code to the right?

A. `20,20.0off`

B. `20,0.2off`

C. `20,20.0% off`

D. `20 off`

E. `,20.0% off`

```
public class Ticket{
  private int price;

  public Ticket(int p){
    price = p;
  }

  public String toString(){
    return price + "";
  }
}

public class StudentTicket extends Ticket{
  private double discount;

  public StudentTicket(int p, double d){
    super(p);
    discount = d;
  }

  public String toString(){
    return super.toString() + "," +
      (discount * 100) + "% off";
  }
}

////////////////////////////////////////
// client code
StudentTicket t;
t = new StudentTicket(20, 0.2);
System.out.print(t);
```

QUESTION 11

What is output by the code to the right?

A. 32 B. 63 C. 64

D. 94 E. 91

```
int ax = 31;
int bx = 63;
int cx = ax ^ bx;
System.out.print(cx);
```

QUESTION 12

What is output by the code to the right?

A. 12 B. 13 C. 13.0

D. 12.0 E. 20

```
double st = 12.99;
System.out.print( Math.round(st) );
```

QUESTION 13

What is output by the code to the right?

A. `Cards` B. `(Cards)` C. `"(Cards)"`

D. `\Cards\` E. `""(Cards)""`

```
String team = "(Cards)";
System.out.print("\"" + team + "\"");
```

QUESTION 14

What is output by the code to the right?

In the answers below ♿ indicates a space.

A. `.35` B. `0.35` C. ♿♿`0.350`

D. `7.3` E. `0.3500000`

```
double value = 0.3500;
System.out.printf("%7.3f", value);
```

QUESTION 15

What is returned by the method call `manip(-2, 2.5)`?

A. `-5.5` B. `-3.5` C. `0.0`

D. `0.5` E. `1.0`

```
public double manip(int x, double a){
  x *= -1;
  a = x - a;
  a++;
  return a;
}
```

QUESTION 16

What is output by the code to the right?

A. `2 2` B. `10 0` C. `10 2`

D. `20 2` E. `0 0`

```
int x2 = 20;
int y2 = 2;
x2 = x2 / (4 / 2);
int x3 = 20 / 4 / y2;
System.out.print(x2 + " " + x3);
```

QUESTION 17

What replaces **<*1>** in the code to the right to get the number of characters in the `String eq`?

A. `eq.index()` B. `new String()`

C. `eq.sub()` D. `eq.numChars()`

E. `eq.length()`

Assume **<*1>** has been filled in correctly.

```
String eq = "3a+2*6-4x";
int tally = 0;
for(int i = 0; i < <*1>; i++){
  char ch = eq.charAt(i);
  if( Character.isLetterOrDigit(ch) )
    tally++;
}
System.out.print(tally);
```

QUESTION 18

What is output by the code to the right?

A. `2` B. `6` C. `0`

D. `3` E. `10`

QUESTION 19

How many `'*'`s are output by the code to the right?

A. `15` B. `30` C. `40`

D. `45` E. `60`

```
for(int i = 0; i < 2; i++){
  for(int j = 0; j < 5; j++) {
    System.out.print('*');
  }
  for(int j = 0; j < 10; j++) {
    System.out.print('*');
  }
}
```

QUESTION 20

Which of the following is a valid Java identifier?

A. `HomeStats` B. `3stars` C. `3_PI` D. `4*5` E. `.<T,K>`

QUESTION 21

What is output by the code to the right?

A. `[2, 3, 3, 5]`

B. `[3, 3, 0, 5]`

C. `[3, 5, 0, 1]`

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

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

```
ArrayList<Integer> exer;
exer = new ArrayList<Integer>();
exer.add(3);
exer.add(5);
exer.add(1, 3);
exer.add(0, 2);
System.out.print(exer);
```

QUESTION 22

What is output by the code to the right?

A. `37.12 5` B. `29 6` C. `19 5`

D. `10 1` E. `49 5`

```
String d1 = "AA8AA 9 10 AA.12 AA 10";
Scanner sc = new Scanner(d1);
int temp = 0;
int tokens = 0;
while(sc.hasNext() ){
  tokens++;
  if(sc.hasNextInt())
    temp += sc.nextInt();
  else
    sc.next();
}
System.out.print( temp + " " + tokens );
```

QUESTION 23

Which of the following Boolean expressions has the truth table to the right?

A. `!p && !q` B. `p || !q`

C. `!q || p` D. `!(p && q)`

E. `p && q`

p	q	result
false	false	true
false	true	false
true	false	false
true	true	false

QUESTION 24

What is returned by the method call `kappa(3, 5)`?

A. `0` B. `5` C. `15`

D. `30` E. `60`

```
public int kappa(int x, int y){
  y *= 2;
  int result = 0;
  for(int i = 0; i < y; i++)
   result += x;
  return result;
}
```

QUESTION 25

What is output by the code to the right when method `mu` is called?

A. `36` B. `42` C. `48`

D. `6` E. `70`

```
public int lambda(double a, double b){
  int x = (int)( a / 0.5 );
  int y =(int)( b / 2 ) + (int)a % 2;
  return kappa(y, x);
}

public void mu(){
  double c = 3.25;
  double d = 4.75;
  System.out.print( lambda(c, d) );
}
```

QUESTION 26

The code to the right contains a syntax error. Which of the following best describes the reason for the syntax error?

A. The statement `days--;` attempts to alter a constant.

B. Constant names must be all capital letters.

C. Variable names may not be all capital letters.

D. Variables of type `double` may not be assigned `int` values.

E. `++WEEKS` must be rewritten as `WEEKS++`.

```
final int days = 365;
double WEEKS = days;
days--;
++WEEKS;
```

QUESTION 27

Which of the following can replace `<*1>` in the code to the right so that the code segment compiles without error?

I. `new Object()`
II. `new ArrayList<String>()`
III. `new Queue<String>()`

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

D. I and II E. II and III

```
List<String> names = <*1>;
```

QUESTION 28

What is output by the code to the right?

A. `0` B. `1`

C. `10` D. `20`

E. The output will vary from one run of the code to the next.

```
ArrayList<Integer> scores;
scores = new ArrayList<Integer>();
System.out.print( scores.size() );
```

QUESTION 29

Method `mysterySort(int[] data)` has the following the following timing data for the arrays shown. Based on this data which sorting algorithm is most likely used by method `mysterySort`?

`data.length`	Time to sort array with all distinct elements in random order.	Time to sort array with all elements equal to the same value.
1,000,000	4 seconds	4 seconds
2,000,000	8.4 seconds	8.4 seconds
4,000,000	17.6 seconds	17.6 seconds

A. binary sort B. insertion sort C. selection sort D. quicksort E. merge sort

QUESTION 30

What is returned by method `find` if `tgt` is `9` and `data` is the array shown below?

`{-5, -1, 3, 5, 12, 13, 20, 100, 101}`

A. `-1` B. `4` C. `5`

D. `9` E. `10`

QUESTION 31

Which of the following best describes what method `find` returns if the precondition is met?

A. Returns the size of `data`.

B. Returns the index in `data` where `tgt` occurs if present, otherwise it returns `-1`.

C. Returns the number of elements in `data` that are greater than `tgt`.

D. Returns the number of elements in `data` that are less than `tgt`.

E. Returns the number of elements in `data` that are equal to `tgt`.

QUESTION 32

Which searching algorithm does method `find` use?

A. Insertion Search B. Sequential Search

C. Selection Search D. Binary Search

E. None of A, B, C, or D are correct.

```
// pre: data != null, elements in data
// are in ascending order, no value
// in data appears more than once
public int find(int[] data, int tgt) {
  int result = -1;
  int s = 0;
  int b = data.length - 1;
  while( result == -1 && s <= b ){
    int m = (s + b) / 2;
    int val = data[m];
    if( val == tgt )
      result = m;
    else if( val < tgt )
      s = m + 1;
    else
      b = m - 1;
  }
  if(result == -1)
    result = s;
  return result;
}
```

QUESTION 33

What is output by method `proc` if `data` initially contains the elements shown below?

`[7, 12, -5, 12, 10]`

A. `[7, 5, -5, 5, 5]`

B. `[5, 12, 5, 12, 10]`

C. `[7, 5, -5, 5, 10]`

D. `[7, 12, -5, 12, 10]`

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

```
public void proc(ArrayList<Integer> data){
  ListIterator<Integer> it;
  it  = data.listIterator();
  while(it.hasNext()){
    if(it.next() < 10)
      it.set(5);
  }
  System.out.print(data);
}
```

QUESTION 34

Assume method `sample(int[] data)` is $O(N^3)$ where N = `data.length`. When method `sample` is passed an array with `length` = 1,000 it takes 1 second for method `sample` to complete. If method `sample` is then passed an array with `length` = 4,000 what is the expected time it will take method `sample` to complete?

A. 2 seconds B. 4 seconds C. 8 seconds D. 16 seconds E. 64 seconds

QUESTION 35

What is output by the code to the right when method `demo` is called if `set1` contains the following values

`{-5, 0, 3, 5, 10}`

and `set2` contains the following values?

`{0, 1, 2, 3, 4, 5}`

A. `false 8 0`

B. `true 11 6`

C. `8 6`

D. `true 8 6`

E. `11 6`

```
public void demo(TreeSet<Integer> set1,
                 TreeSet<Integer> set2) {

  System.out.print(set1.addAll(set2));
  System.out.print( " " + set1.size() );
  System.out.print( " " + set2.size() );
}
```

QUESTION 36

What is output by the code to the right?

A. `true true true`

B. `true true false`

C. `true false true`

D. `false true false`

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

```
Object s1 = "1234";
boolean b1 = s1 instanceof Object;
boolean b2 = s1 instanceof String;
boolean b3 = s1 instanceof Integer;
System.out.print(b1 + " " + b2 + " " + b3);
```

QUESTION 37

Method `aCount` to the right will not compile because the line marked `//error` has the possibility of generating a `FileNotFoundException` which is a type of checked exception. Which of the following changes will allow method `aCount` to compile?

I. Use a `for` loop instead of a `while` loop.

II. Use a `try` and `catch` block.

III. Add a clause to the method header declaring the method `throws a FileNotFoundException`.

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

D. I and II E. II and III

```
public int aCount(String filename) {
  int count = 0;
  File f = new File(filename);
  Scanner sc = new Scanner(f); // error
  while( sc.hasNext() )
    if( sc.next().equals("a") )
      count++;
  sc.close();
  return count;
}
```

QUESTION 35

What is output by the code to the right when method `demo` is called if `set1` contains the following values

`{-5, 0, 3, 5, 10}`

and `set2` contains the following values?

`{0, 1, 2, 3, 4, 5}`

A. `false 8 0`

B. `true 11 6`

C. `8 6`

D. `true 8 6`

E. `11 6`

```
public void demo(TreeSet<Integer> set1,
                 TreeSet<Integer> set2) {

  System.out.print(set1.addAll(set2));
  System.out.print( " " + set1.size() );
  System.out.print( " " + set2.size() );
}
```

QUESTION 36

What is output by the code to the right?

A. `true true true`

B. `true true false`

C. `true false true`

D. `false true false`

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

```
Object s1 = "1234";
boolean b1 = s1 instanceof Object;
boolean b2 = s1 instanceof String;
boolean b3 = s1 instanceof Integer;
System.out.print(b1 + " " + b2 + " " + b3);
```

QUESTION 37

Method `aCount` to the right will not compile because the line marked `//error` has the possibility of generating a `FileNotFoundException` which is a type of checked exception. Which of the following changes will allow method `aCount` to compile?

I. Use a `for` loop instead of a `while` loop.

II. Use a `try` and `catch` block.

III. Add a clause to the method header declaring the method `throws a FileNotFoundException`.

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

D. I and II E. II and III

```
public int aCount(String filename) {
  int count = 0;
  File f = new File(filename);
  Scanner sc = new Scanner(f); // error
  while( sc.hasNext() )
    if( sc.next().equals("a") )
      count++;
  sc.close();
  return count;
}
```

QUESTION 38

Given the `Structure` and `Hold` classes shown to the right, what is output by the following client code?

```
Structure<Integer> s;
s = new Structure<Integer>();
int[] vals = {12, -5, 20, 13, 7};
for(int v : vals)
  s.add(v);
System.out.print( s.access() + " ");
s.remove();
System.out.print( s.access() );
```

A. `7 13`

B. `12 -5`

C. `20 20`

D. `20 13`

E. `-5 7`

QUESTION 39

What type of data structure does the `Structure` class use as its internal storage container?

A. a stack

B. a min heap

C. a binary tree

D. a max heap

E. an array

QUESTION 40

What type of data structure does the `Structure` class implement from client codes' perspective?

A. a stack

B. a priority queue

C. a map

D. a list

E. a hashtable

```
public class Structure<E> {
  private Hold<E> start;

  public void add(E obj) {
    start = addHelp(obj, start);
  }

  private Hold<E> addHelp(E obj,
                          Hold<E> n) {
    if(n == null) {
      n = new Hold<E>();
      n.data = obj;
    } else {
      Comparable c1 = (Comparable)obj;
      Comparable c2 = (Comparable)n.data;
      int val = c1.compareTo(c2);
      if(val < 0)
        n.s1 = addHelp(obj, n.s1);
      else if(val > 0)
        n.s2 = addHelp(obj, n.s2);
    }
    return n;
  }

  public E access(){
    Hold<E> temp = start;
    while( temp.s1 != null )
      temp = temp.s1;
    return temp.data;
  }

  public boolean isEmpty(){
    return start == null;
  }

  public void remove() {
    start = removeHelp(start);
  }

  private Hold<E> removeHelp(Hold<E> n){
    return n.s1 == null ? n.s2 :
      removeHelp(n.s1);
  }
}

public class Hold<E> {
  public E data;
  public Hold<E> s1;
  public Hold<E> s2;
}
```

Standard Classes and Interfaces — Supplemental Reference

class java.lang.Object
- boolean equals(Object other)
- String toString()
- int hashCode()

interface java.lang.Comparable<T>
- 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 Comparable<Integer>
- Integer(int value)
- int intValue()
- boolean equals(Object obj)
- String toString()
- int compareTo(Integer anotherInteger)
- static int parseInt(String s)

class java.lang.Double implements Comparable<Double>
- Double(double value)
- double doubleValue()
- boolean equals(Object obj)
- String toString()
- int compareTo(Double anotherDouble)
- static double parseDouble(String s)

class java.lang.String implements Comparable<String>
- int compareTo(String anotherString)
- boolean equals(Object obj)
- int length()
- String substring(int begin, int end)
  Returns the substring starting at index begin and ending at index (end - 1).
- String substring(int begin)
  Returns substring(from, length()).
- int indexOf(String str)
  Returns the index within this string of the first occurrence of str. Returns -1 if str is not found.
- int indexOf(String str, int fromIndex)
  Returns the index within this string of the first occurrence of str, starting the search at the specified index.. Returns -1 if str is not found.
- charAt(int index)
- int indexOf(int ch)
- int indexOf(int ch, int fromIndex)
- String toLowerCase()
- String toUpperCase()
- String[] split(String regex)
- boolean matches(String regex)

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 a)
- static double abs(double a)
- static double pow(double base, double exponent)
- static double sqrt(double a)
- 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)
- static double random()
  Returns a double value with a positive sign, greater than or equal to 0.0 and less than 1.0.

interface java.util.List<E>
- boolean add(E e)
- int size()
- Iterator<E> iterator()
- ListIterator<E> listIterator()

class java.util.ArrayList<E> implements List<E>
Methods in addition to the List methods:
- E get(int index)
- E set(int index, E e)
  Replaces the element at index with the object e.
- void add(int index, E e)
  Inserts the object e 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 List<E>, Queue<E>
Methods in addition to the List methods:
- void addFirst(E e)
- void addLast(E e)
- E getFirst()
- E getLast()
- E removeFirst()
- E removeLast()

class java.util.Stack<E>
- boolean isEmpty()
- E peek()
- E pop()
- E push(E item)

interface java.util.Queue<E>
- boolean add(E e)
- boolean isEmpty()
- E peek()
- E remove()

class java.util.PriorityQueue<E>
- boolean add(E e)
- boolean isEmpty()
- E peek()
- E remove()

interface java.util.Set<E>
- boolean add(E e)
- boolean contains(Object obj)
- boolean remove(Object obj)
- int size()
- Iterator<E> iterator()
- boolean addAll(Collection<? extends E> c)
- boolean removeAll(Collection<?> c)
- boolean retainAll(Collection<?> c)

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

class java.util.TreeSet<E> implements 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 Map<K,V>

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

interface java.util.Map.Entry<K,V>
- K getKey()
- V getValue()
- V setValue(V value)

interface java.util.Iterator<E>
- 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 e)
- void set(E e)

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)

Computer Science Answer Key
UIL District 2 - 2010

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

Notes:

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.

29. Both cases exhibit O(NlogN) behavior with the time to process double the number of elements increasing by slightly more than double. Mergesort is O(NlogN) in the average and worst case. The traditional quicksort is O(NlogN) in the average case, but degenerates to $O(N^2)$ in the worst case. The worst case quicksort occurs if the pivot is always in the min or max and this occurs if all elements are equal.