Computer Science
Tompkins High School
March 8th 2014

Directions:

1. DO NOT OPEN THE EXAM UNTIL INSTRUCTED TO DO SO.

2. NO CALCULATORS or calculation devices may be used during the exam.

3. You will have 45 minutes to complete the exam.

4. When time is called you may finish writing down a letter if it is already started.

5. When you are finished with your exam wait quietly.

1. What is the result of the following expression?

$17_8 + 2F_{16}$

A. $3E_{16}$ B. 60_{10} C. 101110_2 D. 75_8 E. $3D_{16}$

2. What is output by the code to the right?

A. 14 B. 7.5
C. 7 D. 17.5
E. 17

```
int a = 7;
int b = 14;
int c = a%b +a/b;

System.out.println(c);
```

3. What is output by the code to the right?

A. 16 9
B. 18 10
C. 18 11
D. 17 11
E. 16 10

```
int a = 7;
int b = a+3;
a =a+b++;

System.out.println(a + " " + b);
```

4. What is output by the code to the right?

A. 52
B. 56
C. 45
D. 64
E. 60

```
int total = 0;
for(int a = 0; a<=3; a++)
{
   for(int b = a; b<=5; b++)
   {
      total+=b;
   }
}
System.out.println(total);
```

5. What is output by the code to the right?

A. tetextppend
B. texexttappend
C. tetextappend
D. texttexttextappend
E. Runtime Exception

```
StringBuffer text = new StringBuffer("texttexttext");
text.append("append");
text.delete(3,5);
text.delete(7,10);

System.out.println(text);
```

6. What is output by the code to the right?

A. a b 3 B. 3 b a
C. 51 66 49 D. 49 66 51
E. c B a

```
int[][] a = { {'a','7','3'},
              {'2','B','C'},
              {'1','2','c'}};

for(int x=0; x<a.length;x++)
       System.out.print((int)a[x][2-x] + " ");
```

7. What values of c would make a true?

A. (50, 55]
B. (50, 55)
C. (50, 55]
D. [50, 55)
E. None of the above

```
boolean a = !(50>c || 55<=c);
```

8. What is output by the code to the right?

A. BD
B. BCD
C. BC
D. ACD
E. ABD

```
int y = 1500;

if(y%400==0)
   System.out.print("A");
if(y%100==0)
   System.out.print("B");
if(y%4!=0)
   System.out.print("C");
else
   System.out.print("D");
```

9. What is output by the code to the right?

A. *768.988 * B. * 768.988*
C. *-768.988* D. *-11534.814*
E. Syntax Error

```
System.out.printf("*%-10.3f*",768.98765);
```

10. What is output by the <*1> line?

A. false false
B. false true
C. true false
D. true true
E. false

11. What is output by the <*2> line?

A. false false
B. false true
C. true false
D. true true
E. false

12. What is output by the <*3> line?

A. 13
B. 319
C. 519
D. 15
E. 17

```
public class ClassA{
   public int a;
   public ClassA(int a)
   {   this.a = a;   }

   public ClassA()
   {  a=3;   }
}

public class ClassB extends ClassA{
   public int b;
   public ClassB(int a,int b)
   {
      super(a);
      this.b = b;
   }

   public ClassB()
   {
      super(1)
      b=9;
   }
}

public class MainClass{
   public static void main(String[] args)
   {
      ClassA m = new ClassA(5);
      ClassA n = new ClassB();

      System.out.println(m instanceof ClassA + " "
+ m instanceof ClassB); //<*1>

      System.out.println(n instanceof ClassA + " "
+ n instanceof ClassB); //<*2>

      System.out.println(m.a + n.a +n.b); //<*3>
    }
}
```

13. What is output by the code to the right?

A. ADCFF B. AFCAD
C. DACFA D. ACFAD E. Runtime Exception

```
char[] letters = {'A','D','C','F','B'};

for(char a:letters)
   System.out.print(letters[a%5]);
```

14. What is output by <*1> in the code to the right?

A. 1 B. 0
C. -1 D. 2
E. 3

15. What is output by <*2> in the code to the right?

A. 1 B. 0
C. -1 D. 2
E. 3

16. What is output by <*3> in the code to the right?

A. 1 B. 0
C. -1 D. 2
E. Runtime Error

```
public static int methodA(char[][] b,int c)
{
   for(int r=b.length-1; r>=0;r--)
      if(b[r][c]=='-')
         {
            b[r][c] = '*';
            return r;
         }
   return -1;
}

public static void main(String[] args)
{
   char[][] b= {  {'-','-','-','*'},
                  {'-','-','*','*'},
                  {'-','*','*','*'}};
   methodA(b,2);
   System.out.println(methodA(b,2)); //<*1>
   System.out.println(methodA(b,1)); //<*2>
   System.out.println(methodA(b,4)); //<*3>
}
```

17. What is output by the code to the right?

A. 3 4 8 B. 3 4 6
C. 3 4 8 D. 3 3 6
E. 5 3 6

```
int x =3, y=4, z=3;

if(x>--y && x++ <6)
   z= 8;
else if(y%3==0 || ++x >=17)
   z = 6;
System.out.println(x + " " + y + " " + z);
```

18. What is the result of the expression to the right?

A. 30 B. 17
C. 34 D. 16
E. 32

```
 17>>2<<3
```

19. What is output by the code to the right?

A. 3 B. 56
C. 54 D. 14
E. A5

```
int x = 40 | 0b1110;
System.out.println(Integer.toString(x,8));
```

20. What is output by the code to the right?

A. ZSM
B. ZSS
C. MSZ
D. MSS
E. Runtime Error Error

```
Stack<String> letters = new Stack<String>();
letters.push("M");
letters.push("S");
letters.push("Z");

System.out.println(letters.pop()+letters.peek()
+letters.pop());
```

21. What is output by <*1> in the code to the right?

A. Bob
B. Joe
C. JoeBob
D. BobJoe
E. JBooeb

22. What is output by <*2> in the code to the right?
A. BobBobJoe
B. JoeBobJoe
C. BobJoeBob
D. JoeJoeBob
E. JoeJoeJoe

23. What is output by <*3> in the code to the right?
A. Bob
B. Joe
C. JoeBob
D. BobJoe
E. JBooeb

```
public class Test1{
   public void text()
   {   System.out.println("Bob");   }
}

public class Test2 extends Test1{
   public void text()
   {   System.out.println("Joe");   }

   public void doubleText()
   {   test() + super.test() + this.text();   }
}

public class MainClass{
   public static void main(String[] args)
   {
      Test1 a = new Test1();
      Test2 b = new Test2();
      a.text(); //<*1>
      b.doubleText() ; //<*2>
      b.text(); //<*3>
    }
}
```

24. What values would make b true?

A. 18 17 3
B. 22 26 28
C. 2800 1900 24
D. 2600 2800 3
E. 2800 1904 2400

```
boolean b=false;

if(y%4==0)
{
  if(y%100==0)
  {
     if(y%400==0)
        b=true;
     else
        b= false;
  }
  else
     b= true;
}
else
  b= false;
```

25. What is the big of the code to the right?

A. O(1) B. O(n/2)
C. O(n log n) D. O(log n)
E. O(n)

```
for(int a=a/2; a>=0; a=(a-1)/2)
   System.out.println(a);
```

26. What replaces <*1> in the code to the right so that data is correctly sorted after the following call?

```
    sortA(data);
```

A. i+1
B. i
C. j+1
D. j-1
E. i-1

```
public void sortB(int[] data)
{
   int n = data.length;
   int j = 0;
   for(int i = 1; i < n; i++)
   {
      int temp = data[i];

      for(j=i; j > 0 && data[j-1] > temp; j--)
         data[j] = data[<*1>];
      data[j] = temp;
    }
}
```

27. What type of sort does the code to the right implement?

A. Insertion B. Quick
C. Merge D. Selection
E. Heap

28. What is output by the code to the right?

A. 2 B. 33 C. 41 D. 1
E. 46

```
int a = 6;
int b = 35;
int c = (a>b)?b+c:(b==a)?b-8&2:7-a|b;
System.out.println(c^13);
```

29. What is output by the code to the right?

A. Bill 2 []
B. Bill 2 [Bill 2]
C. Joe 5 []
D. Joe 5 [Joe, 5]
E. Bill 2 [Joe, 5]

```
public static void test(String a, Integer b,
   ArrayList<String> c)
{
    b=5;
    a="Joe";
    c.add(a);
    c.add(b.toString());
}

public static void main(String[] args)
{
    ArrayList<String> data = new ArrayList<String>();
    String a = "Bill";
    Integer b = 2;
    test(a,b,data);
    System.out.println(a + " "+ b + " " + data);
}
```

30. What sort completes the faster when the data set is already sorted?

A. Selection
B. Insertion
C. Quick Sort
D. Merge Sort
E. Heap Sort

31. What is output by the code to the right?

A. [18, 2, 27, 4]
B. [18, 2, 28, 5]
C. [19, 3, 28, 5]
D. [19, 3, 27, 4]
E. [54, 6, 84, 14]

```
public static void shrink(int[] data, int change)
{
   for(int x=0; x<data.length;x++)
      data[x]= data[x]/change;
}

public static void main(String[] args)
{
   int[] data = {54,6,83,14};
   shrink(data,3);
   System.out.println(Arrays.toString(data));
}
```

32. What is the result of the recur("ABCD")?

A. ABCDCEBDDC
B. ABCDABCDABC
C. ABCDCEBDDCC
D. ABCDDCBAAB
E. ABCDCBDCAB

33. What is the result of the recur("DIG")?

A. DIGFJIEKHFJ
B. DIGFJIEKHF
C. DIGDIGDIGD
D. DIGFJICGHA
E. DIGHJKWLZS

```
public static String recur(String a)
{
   if(a.length() >10)
      return a;
   else if(a.length()%2==0)
      return recur(a+(char)(a.charAt(a.length()%3)+1));
   else
      return recur(a+(char)(a.charAt(a.length()%3)+2));
}

public static void main(String[] args)
{
   System.out.println(recur("ABCD"));
   System.out.println(recur("DIG"));
}
```

34. Assuming h is a minimum heap built with an ArrayList what would be its ArrayList look like after the code to the right has been executed?

A. [1, 3, 5, 6]
B. [6, 5, 3, 1]
C. [1, 3, 6, 5]
D. [1, 5, 3, 6]
E. [6, 3, 1, 2]

```
h.add(5);
h.add(3);
h.add(6);
h.add(1);
```

35. Assuming h is a minimum heap built with an ArrayList what would be its ArrayList look like after the code to the right has been executed?

A. [-1, 3, 12, 17]
B. [17, 12, 3, -1]
C. [17, 3, 12, -1]
D. [-1, 12, 3, 17]
E. [-1, 12, 17, 3]

```
h.add(3);
h.add(2);
h.add(12);
h.remove();
h.add(1);
h.remove();
h.add(-1);
h.add(17);
```

36. What replaces <*1> in the code to the right so add works correctly for a maximum heap built with an ArrayList?

A. (index-2)/2
B. (index-1)/2
C. index-2
D. (index*2)+1
E. (index*2)+2

```
public boolean add(E item)
{
   data.add(item);
   int current = data.size()-1;

   while(current!=0 &&
data.get(current).compareTo(data.get(<*1>)) < 0)
   {
      swap(current,<*1>);
      current = <*1>;
   }
   return true;
}
```

37. What is output by the code to the right?

A. 32
B. -27
C. 27
D. -32
E. 2

```
String a = "Bottles";
String b = "bottom";
System.out.println(a.compareTo(b));
```

38. What is output by the code to the right?

A. The d*a*ons a*e hun*y*
B. The d*a*ons a*e hun**y*
C. The dragons are hun*y*
D. The dragons are hun**y*
E. The dragons are hun**y**

```
String s = "The dragons are hungry";
String[] parts = s.split("[gr]");

for(String a: parts)
   System.out.print(a+"*");
```

39. What is output by the code to the right?
 	
A. 15
B. 7
C. 31
D. 62
E. 251
	`int a = 31;`
	`int b = (a<<3)|(a>>3);`
	
	`System.out.print(b);`

40. What is the range of possible numbers a can store after the code to the right is executed? A. [5,11] B. [5,11) C. [4,10] D. [4,10) E. None of the Above	`int a=(int)Math.random()*5+6;`

Standard Classes and Interfaces — Supplemental Reference
(Accessed From: UIL COMPUTER SCIENCE DISTRICT 2 2011)

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()
- 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.ArrayList<E> implements List<E>
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.lang.Exception

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)