A+ Computer Science
M/C Written Test

General Directions:

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

2) **NO CALCULATORS of any kind may be used.**

3) 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 forty-five 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 except on the answer sheet or Scantron card which is 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 (i.e. `error` is an answer choice). Ignore any typographical errors and assume any undefined variables are defined as used.**

9) A reference to commonly used Java classes is provided with the test and you may use this reference 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 23 Packages and classes (e.g. .lang, .util, System, Math, Double, etc.) are included in any programs or code segments that refer to methods from these classes and/or 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 each incorrect answer.

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()`
- `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.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)

A+ Computer Science Contest #2425-13 February 15, 2025

Note: Correct responses are based on **Java SE Development Kit 23 (JDK 23)** from Oracle, Inc. All provided code segments are intended to be syntactically correct, unless otherwise stated (e.g., "error" is an answer choice) and any necessary Java SE 23 Standard Packages have been imported. Ignore any typographical errors and assume any undefined variables are defined as used. **For all output statements, assume that the System class has been statically imported using: import static java.lang.System.***

QUESTION 1

Which of the following is equivalent to 45_{12} ?

A. 53_{10} B. 58_{9} C. 105_{7} D. 124_{6} E. 144_{8}

QUESTION 2

What is output by the code to the right?

A. 14 B. 17 C. 10 D. 12

E. There is no output due to an error.

```
out.println(2 * 5 + 7 / 3);
```

QUESTION 3

What is output by the code to the right(i's indicate blank spaces)?

A. 2,345.5

B. 2,345.488

C. 002,345.5

D. 002345.488

E. 0,002,345.5

```
out.printf("%,09.1f",2345.488);
```

QUESTION 4

What is output by the code to the right?

A. kast
B. KAST
C. KASTE
D. kaste
E. There is no output due to an error.

```
String s = "OutKasted";
s = s.toUpperCase();
s = s.substring(3, 7);
out.println(s);
```

QUESTION 5

What is output by the code to the right?

A. True B. False C. false D. true

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

```
boolean a = false;
boolean b = true;
a = b ^ a ^ b ^ b;
b |= a & b | a;
out.println(b);
```

QUESTION 6

What is output by the code to the right?

A. 6.5 B. 6.0 C. 7.0 D. 6.428571

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

```
int a = 45, b = 7;
double c = Math.floorDiv(a, b);
out.println(c);
```

QUESTION 7

What is the output by the code to the right?

A. 2 B. 13

C. 3 D. 12

E. 1

```
int j = 8;
if(j <= 8)
  out.print(1);
else if(j >= 8)
  out.print(2);
else
  out.print(3);
```

QUESTION 8

What is the output by the code to the right?

A. 64
B. 127
C. 255
D. 128
E. 256

```
int sum = 1;
for(int y = 1; y < 103; y += y)
  sum += y;
out.println(sum);
```

QUESTION 9

What is output by the code to the right?

A. 3
B. 8
C. 5
D. 12
E. There is no output due to an error.

```
int a = 5 * 2 & 9 + 12 | 3;
out.println(a);
```

QUESTION 10

What is the output by the code to the right ?

A. 2bY
B. 1Xa
C. 1aX
D. 2Yb
E. There is no output due to an error.

```
char[] c = "ZYX123abc".toCharArray();
Arrays.sort(c);
String s = ""+c[1]+c[4]+c[7];
out.println(s);
```

QUESTION 11

What is the output by the code to the right?

A. 210 4
B. 186 4
C. 185 4
D. 220 4
E. There is no output due to an error.

```
String s = "12 32 43 54 ";
s += "99 12 89";
int sum = 0, cnt = 0;
Scanner sc = new Scanner(s);
while(cnt < 4) {
  cnt++;
  if(sc.nextInt() % 2 == 0)
     sum += sc.nextInt();
}
out.println(sum+" "+cnt);
```

QUESTION 12

What is the output by the code to the right(^'s represent spaces)?

A. 12
B. 6
C. 15
D. 8
E. There is no output due to an error.

```
int[] i = new int[] {
  3, 6, 2, 6, 4, 9, -2};
for(int y = 0; y < 5; y++)
  i[y] += i[y + 2];
out.println(i[2] + i[4]);
```

QUESTION 13

What is the order of precedence for the operators to the right ?

A. I, IV, III, II
B. IV, I, II, III
C. I, IV, II, III
D. IV, I, III, II
E. I, II, III, IV

```
I. *
II. >=
III. ==
IV. - (subtractive)
```

QUESTION 14

What is the output by the code to the right ?

A. 6464
B. 3264
C. 1616
D. 1632
E. 3232

```
int[] i = new int[5];
i[0] = Double.SIZE;
i[1] = Short.SIZE;
i[2] = Byte.SIZE;
i[3] = Integer.SIZE;
i[4] = Float.SIZE;
Arrays.sort(i);
out.println(i[2]+""+i[3]);
```

QUESTION 15

What is the output by the code to the right ?

A. `[Green, Purple, Orange]`

B. `[Red, Green, Purple, Blue]`

C. `[Red, Green, Orange]`

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

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

```
ArrayList<String> a;
a = new ArrayList<String>();
a.add("Purple");
a.add("Blue");
a.add(0, "Red");
a.add(2, "Orange");
a.set(1, "Green");
a.remove(3);
out.println(a);
```

QUESTION 16

What is output by the code to the right ?

A. `4 8 1`

B. `4 8 -7`

C. `5 7 3`

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

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

```
int a = 5;
int b = 7;
int c = 10;
c = b+++-c--+--a;
out.println(a + " " + b + " " + c);
```

QUESTION 17

Which of the following could replace **<1*>** in the code to the right so that it will compile without error?

A. `(byte)`

B. `(int)`

C. `(long)`

D. A and B only

D. A, B, and C

```
final byte a = 3;
final byte b = 8;
byte c = <1*>(a + b);
```

QUESTION 18

What is output by the code to the right ?

A. `[3, 4]`

B. `[1, 2]`

C. `[3, 4, 2]`

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

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

```
ArrayList<Integer> a;
a = new ArrayList<Integer>();
a.add(3);
a.add(1);
a.add(4);
a.add(2);
a.remove(new Integer(3));
a.remove(1);
out.println(a);
```

QUESTION 19

What is output by the code to the right ?

A. `true true`

B. `false true`

C. `true false`

D. `false false`

E. There is no output due to an error.

```
double x = 1.0/0.0;
double y = 0.0/0.0;
out.print(x == x + 1);
out.print(" ");
out.println(y != y);
```

QUESTION 20

Which of the following is not a valid Java identifier?

A. `Ca$HMoney` B. `Hi_There` C. `Question?` D. `H3LL0` E. All are valid.

QUESTION 21

What is output by the code to the right?

A. `false true`
B. `true true`
C. `true false`
D. `false false`
E. There is no output due to a runtime error.

```
String s1 = "What R U Hiding ";
String s2 = "Who will U B Tonight ";
String m = "([A-Z]\\w* ){3,5}";
String s = "" + s1.matches(m);
s += " " + s2.matches(m);
out.println(s);
```

QUESTION 22

Which of the following could replace **<1*>** in the code to the right so that any child of the `Bird` class must define method `noise()`?

A. `protected`
B. `abstract`
C. `implementable`
D. `extendable`
E. More than one of the above.

QUESTION 23

Which of the following could replace **<2*>** in the code to the right so that the `toString()` method is defined without error, and will output `"Caw!! "`, followed by the value of the instance variable `speed`?

A. `"Caw!! speed"`
B. `"Caw!! " + speed`
C. `noise() + " " + speed`
D. Both B and C

QUESTION 24

Which of the following could replace **<3*>** in the code to the right so that the `Hawk` is a child of both `Flyer` and `Bird`?

A. `extends Flyer, Bird`
B. `extends Flyer implements Bird`
C. `extends Bird implements Flyer`
D. `implements Flyer, Bird`
E. More than one of the above.

QUESTION 25

Which of the following could replace **<4*>** in the code to the right so that the `fly(int s)` is defined without error?

A. `public`
B. `protected`
C. `private`
D. A and B.
E. All of the above.

QUESTION 26

Assuming **<1*>, <2*>, <3*>** and **<4*>** have been filled correctly, what is output by the client code to the right?

A. `12 45 - Caw!!!`
B. `47 - Caw!!!`
C. `12 47 - Caw!!`
D. `45 - Caw!!!12`
E. There is no output due to an error.

```
abstract class Bird{

 protected int speed;

 <1*> String noise();

 public String toString() {
  return <2*>;
 }
}

interface Flyer{

 int fly(int s);

}

class Hawk <3*>{

 private int height;

 <4*> int fly(int s) {
  speed += s;
  return speed;
 }

 public String noise() {
  return "Caw!!!";
 }

 public void fly() {
  height++;
  out.print(height);
 }

}

//////////////client code/////////////
Hawk a = new Hawk();
a.fly(10);
a.fly();
a.fly(25);
a.fly();
String o = " " + a.fly(10);
o += " - " + a.noise();
out.println(o);
```

QUESTION 27

A selection sort agorithm (assume average-case) takes 64 seconds to process an array of length 16,000. How long will the exact same algorithm take to process an array of length 4,000?

A. 16 seconds.
B. 4 seconds.
C. 1 second.
D. 2 seconds.
E. 8 seconds.

QUESTION 28

What is output by the line marked `//q28` in the client code to the right?

A. `55`
B. `41`
C. `73`
D. `31`
E. There is no output due to a runtime error.

QUESTION 29

What is output by the line marked `//q29` in the client code to the right?

A. `171`
B. `301`
C. `399`
D. `227`
E. There is no output due to a runtime error.

```
public int recur(int r) {
  if(r <= 0)
    return 1;
  int i = recur(r - 3);
  i += recur(r - 2);
  return 1 + i;
}
//////////////client code//////////////
out.println(recur(10)); //q28
out.println(recur(16)); //q29
```

QUESTION 30

What is output by the code to the right ?

A. `Orange`
B. `Purple`
C. `Magenta`
D. `Black`
E. There is no output due to an error.

```
TreeMap<String,String> tm;
tm = new TreeMap<String,String>();
tm.put("Blue", "Red");
tm.put("Purple", "Green");
tm.put("Red", "Black");
tm.put("Yellow","Blue");
tm.put("Green","Orange");
tm.put("White", "Magenta");
tm.put("Orange","Brown");
tm.put("Black","Purple");
tm.put("Brown","White");
tm.put("White","Yellow");
String cur = "Orange";
int num = 123;
while(num-- > 0 && tm.containsKey(cur))
  cur = tm.get(cur);
out.println(cur);
```

QUESTION 31

What is output by the code to the right

A. `[94, 212, 123, 450]`
B. `[94, 123, 212, 450]`
C. `[37, 17, -8, -75]`
D. `[212, 94, 123, 450]`
E. There is no output due to an error.

```
PriorityQueue<Integer> pq;
pq = new PriorityQueue<Integer>();
pq.add(212);
pq.add(17);
pq.add(-8);
pq.add(123);
pq.add(94);
pq.poll();
pq.add(-75);
pq.add(450);
pq.poll();
pq.poll();
pq.add(37);
pq.poll();
out.println(pq);
```

QUESTION 32

What could replace **<1*>** in the code to the right so that the `resize()` method works as intended, resizing the array?

A. `System.arraycopy(arr, 0, copy, 0, n);`

B. `System.arraycopy(copy, 0, arr, 0, n);`

C. `System.arraycopy(arr, n, copy, n, 0);`

D. `System.arraycopy(copy, n, arr, n, 0);`

E. `System.arraycopy(arr, 0, copy, 0, m);`

QUESTION 33

What is output by the line marked `//q33` in the client code to the right?

A. `63 10`
B. `127 11`
C. `11 127`
D. `11 63`
E. `63 10`

QUESTION 34

What is output by the line marked `//q34` in the client code to the right?

A. `5 -12`
B. `10 -23`
C. `5 11`
D. `10 22`
E. `10 -12`

QUESTION 35

What is output by the line marked `//q35` in the client code to the right?

A. `Money A null`

B. `Money Made Lasagna`

C. `Money Made null`

D. `Money Being Silence`

E. `Money Being In`

QUESTION 36

Which of the following data structures is implemented by the `Structure` class to the right ?

A. `Min-Heap`

B. `Max-Heap`

C. `Stack`

D. `Queue`

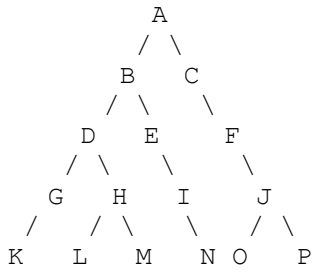
E. `Binary-Search Tree`

```
class Structure{
 Comparable[] arr;
 int size, n;
 public Structure() {
  size = 0;
  n = 1;
  arr = new Comparable[1];
 }
 public void add(Comparable i) {
  if(size == arr.length)
   resize();
  int cur = 0;
  if(size == 0)
   arr[0] = i;
  else {
   while(arr[cur] != null) {
    int j = i.compareTo(arr[cur]);
    if(j <= 0)
     cur = (cur + 1) * 2 - 1;
    else
     cur = (cur + 1) * 2;
    if(cur >= n)
     resize();
   }
   arr[cur] = i;
  }
  size++;
 }
 public void resize() {
  int m = arr.length * 2 + 1;
  Comparable[] copy = new Comparable[m];
  <1*>;
  n = m;
  arr = copy;
 }
 public int find(Comparable i) {
  int cur = 0;
  while(cur < n && arr[cur] != null) {
   int j = i.compareTo(arr[cur]);
   if(j < 0)
    cur = (cur + 1) * 2 - 1;
   else if(j > 0)
    cur = (cur + 1) * 2;
   else
    return cur;
  }
  return -1 - cur;
 }
}
//////////////client code//////////////
Structure s = new Structure();
String[] t = {"Money", "Moves", "Are",
  "Being","Made","In","Silence",
  "Like","A","Lasagna","Chief"};
for(String r:t)
 s.add(r);
out.println(s.size+" "+s.n); //q33
String s1 = "Made", s2 = "Make";
String r = "" + s.find(s1);
r+= " " + s.find(s2);
out.println(r); //q34
r = "" + s.arr[0];
r += " " + s.arr[10];
r += " " + s.arr[17];
out.println(r); //q35
```

QUESTION 37

What is the diameter of the tree to the right?

A. 9
B. 8
C. 5
D. 6
E. 7



QUESTION 38

What is the pre-order traversal of the tree to the right?

A. ABCDEFGHIJKLMNOP
B. KGDLHMBNIEACFOJP
C. KGLMHDNIEBOPJFCA
D. PJOFCAEINBMHLDGK
E. ABDGKHLMEINCFJOP

QUESTION 39

What is the most simplified boolean expression that generates the following truth table?

A	B	C	Result
0	0	0	0
0	0	1	0
0	1	0	1
0	1	1	1
1	0	0	1
1	0	1	1
1	1	0	0
1	1	1	1

QUESTION 40

What is the 8-bit two's complement value corresponding to the following number?

-105_{10}