THE VIRTUAL MEET EXPERIENCE

2024-2025

HS VCM #5 - THE RQ MEET

COMPUTER SCIENCE

DO NOT OPEN TEST UNTIL TOLD TO DO SO

The Virtual Challenge Meets™

2024-2025 High School
VCM #5 - The RQ Meet

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)

2024-2025 Virtual Challenge Meet #5 - The RQ Meet – Computer Science

Note: Correct responses are based on **Java SE Development Kit 23 (JDK 23)**, from Sun Microsystems, Inc. All provided code segments are intended to be syntactically correct, unless otherwise stated (i. e. `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

What is 10001101000_2 plus 631_{10} ?

A. $4AC_{16}$ B. $6DF_{16}$ C. 11011101111_2 D. 1757_{10} E. 11011011011_2

QUESTION 2

What is output by the code to the right?

A. 1 B. 11 C. 5
D. 21 E. 165

```
int x = 16;
x = x + 5;
out.println(x);
```

QUESTION 3

What is output by the code to the right?

A. 212
B. 211
C. 23
D. 5
E. 2121

```
out.println("21" + 2);
```

QUESTION 4

What is output by the code to the right?

A. 0
B. 25
C. 4
D. 5
E. 10

```
String x;
x = "aplusaplusaplusaplusaplus";
x.replace("plus","a");
x.replace("aaaa","x");
out.println(x.length());
```

QUESTION 5

What values for a, b, and c make the output to the right `true`?

A. a is true, b is false, c is true
B. a is true, b is false, c is false
C. a is true, b is true, c is true
D. a is false, b is true, c is false
E. A and B

```
boolean a, b, c, d;
d = !b && a || b && !a && c;
out.println(d);
```

QUESTION 6

What is output by the code to the right?

A. 3
B. 4
C. 3.0
D. 4.0
E. 0

```
out.println(Math.round(3.3));
```

QUESTION 7

What is output by the code to the right?

A. 20
B. 22
C. 19
D. 21
E. 17

```
int sum = 5;
for(int i=25; i<87; i+=10)
  sum+=2;
out.println(sum);
```

QUESTION 8

What is output by the code to the right?

A. 53
B. 50
C. 55
D. 51
E. 52

```
int x = 50;
if(x>50)
x++;
if(x>50)
x++;
if(x>50)
x++;
out.println(x);
```

QUESTION 9

How many lines are printed by the code to the right?

A. 32
B. 23
C. 14
D. 15
E. 16

```
String s = "";
s = "practice.apluscompsci.com";
s = "apluscompsci.com";
for(int i=s.length(); i>1; i--)
   out.println(s.substring(i-1));
```

QUESTION 10

What is output by the code to the right?

A. 35 B. 11 C. 37 D. 44 E. 5

```
int[] list={17,19,5,35,11,37,44,23};
out.print(list[4]);
```

QUESTION 11

Which of the following correctly replaces **<*1>** in the code to the right so that the code would sum **ALL** values in String y?

A. `input.hasNext()`

B. `input.hasNextDouble()`

C. `input.hasNextInt()`

D. A and B only

E. A, B, and C

```
long x = 0;
String y = "25 36 4277669987554 15 1";
Scanner input = new Scanner(y);
while(  <*1>  )
  x += input.nextLong();
out.println(x);
```

QUESTION 12

What is output by the code to the right?

A. `19`

B. `23`

C. `17`

D. `16`

E. `18`

```
int[] x = {3,1,2,3,1,4};
int i=4;
int count = 0;
while(x[i]!=0)
{
  count+=x[i];
  i=(i+x[i]--)%x.length;
}
out.println(count);
```

QUESTION 13

Which of the following has the lowest precedent in java?

A. `&`

B. `&&`

C. `+` (String concatenation)

D. `>>`

E. `<`

QUESTION 14

What is output by the code to the right?

A. `-128`

B. `-32768`

C. `0`

D. `-1`

E. `-65535`

```
out.println(Short.MIN_VALUE);
```

QUESTION 15

What is output by the code to the right?

A. `1`

B. `2`

C. `0`

D. `-1`

E. `4`

```
ArrayList<Integer> list;
list = new ArrayList<>();
list.add(4);
for(int i=0; i<list.size(); i++)
   list.add(list.remove(i));
out.println(list.size());
```

QUESTION 16

What is output by line **//1** in the code to the right?

A. 4
B. null
C. 2
D. 8
E. 9

QUESTION 17

What is output by line **//2** in the code to the right?

A. 4
B. null
C. 2
D. 8
E. 9

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

m.put(3,4);
m.put(5,8);
m.put(2,9);

out.println( m.put(3,2) );   //1

out.println( m.get(4) );   //2
```

QUESTION 18

What is output by line **//1** in the code on the right?

A. 57
B. 45
C. 151
D. 47
E. 150

QUESTION 19

What is output by line **//2** in the code to the right?

A. 60
B. 9
C. 46
D. 63
E. 45

```
int[][] mat = {{2,3,2,0},
               {3,4,4,5},
               {1,4,9,4},
               {2,1,2,1}};

int sum = 0;
for(int i=0; i<mat[0].length; i++)
  for(int j=0; j<mat.length; j++)
  {
     sum+=mat[j][i];
     mat[j][i]=sum;
  }

out.println(sum);          //1
out.println(mat[2][3]);   //2
```

QUESTION 20

What is output by the code to the right?

A. 8
B. 240
C. 248
D. 0
E. 200

```
int x = 48;
int y = 38;
int z = x << 2 | y << 1 & x - y;
out.println(z);
```

QUESTION 21

What is stored in list after the method call `mys1(list,false,3)` if `list` was defined as `list[] = {2,6,4,3,5,8};`?

A. `[0, 2, 0, 0, 3, 7]`

B. `[0, 4, 2, 1, 3, 6]`

C. `[0, 4, 2, 1, 4, 8]`

D. `[1, 3, 1, 0, 3, 7]`

E. There is no output due to a run-time error

QUESTION 22

What is returned by the method call `mys1(list,false,3)`if `list` was defined as `list[] = {2,6,4,3,5,8};`?

A. `17`

B. `19`

C. `13`

D. `16`

E. There is no output due to a run-time error

```
public static int mys1(int[] list,
                       boolean x, int t)
{
  int s=0;
  int k;
  k = (x)?1:-1;
  while(list[t]>0)
  {
   if((t==list.length-1 && x)||
      (t==0 && !x))
   {
    x=!x;
    s++;
    k*=-1;
   }
   list[t]--;
   t+=k;
   s++;
  }
  return s;
}
```

QUESTION 23

How many methods at minimum must class `Not` have?

A. `0`

B. `1`

C. `2`

D. `3`

E. `4`

QUESTION 24

As defined in interface `GONE`, method `go` has which type of access?

A. `private`

B. `public`

C. `protected`

D. A and B only

E. A, B, and C

QUESTION 25

When method `go` is implemented in `Not`, it could be defined with which type of access?

A. `private`

B. `public`

C. `protected`

D. A and B only

E. A, B, and C

```
public interface GONE
{
   int go();
   public void it( int x );
}

public class Not implements GONE
{
    //?????????????
}
```

QUESTION 26

What is output by the code to the right?

A. `212`

B. `211`

C. `-212`

D. `-211`

E. `0`

```
out.println(~211);
```

QUESTION 27

What is output by line **`//1`** in the code to the right?

A. `[1, 2, 3, 3, 3, 6, 18, 8, 4, 4]`

B. `[1, 2, 3, 3, 3, 6, 18, 4, 8, 4]`

C. `[1, 2, 3, 6, 3, 3, 18, 8, 4, 4]`

D. `[1, 2, 3, 3, 3, 6, 4, 4, 8, 18]`

E. `[1, 2, 4, 3, 3, 6, 18, 8, 3, 4]`

QUESTION 28

What is output by line **`//2`** in the code to the right?

A. `[3, 3, 4, 4, 3, 6, 18, 8]`

B. `[3, 3, 3, 4, 4, 6, 8, 18]`

C. `[3, 3, 3, 4, 4, 8, 18, 6]`

D. `[3, 3, 3, 6, 4, 4, 18, 8]`

E. `[3, 3, 3, 4, 4, 6, 18, 8]`

QUESTION 29

What is output by line **`//3`** in the code to the right?

A. `[3, 6, 4, 8, 4, 18]`

B. `[3, 4, 4, 8, 6, 18]`

C. `[3, 4, 4, 6, 8, 18]`

D. `[3, 4, 6, 8, 4, 18]`

E. `[3, 4, 4, 8, 18, 6]`

```
PriorityQueue<Integer> pq;
pq = new PriorityQueue<>();

int[] list = {3,8,6,1,3,3,18,4,2,4};

for(int x:list)
  pq.add(x);

out.println(pq);   //1

pq.remove();
pq.remove();

out.println(pq);   //2

pq.remove();
pq.remove();

out.println(pq);   //3
```

QUESTION 30

What is output by the code to the right?

A. `13`

B. `21`

C. `22`

D. `11`

E. `16`

```
int x = 4;
out.println(12^x^11^19);
```

QUESTION 31

What is output by the code to the right?

A. `298`

B. `240`

C. `248`

D. `170`

E. `291`

```
out.println(0x123);
```

QUESTION 32

What is output by the code on the right?

A. `6`

B. `18`

C. `14`

D. `8`

E. There is no output due to a run-time error

```
int[] x = {20, 29, 22, 18, 38, 41};
Stack<Integer> a = new Stack<>();
Stack<Integer> b = new Stack<>();
int i = 0;
for(int n:x) {
     while(!a.isEmpty()&&a.peek()>n){
        b.push(a.pop());
        i++;
     }
     a.push(n);
     while(!b.isEmpty()){
        a.push(b.pop());
        i++;
     }
}
out.println(i);
```

QUESTION 33

What is returned by `go(8)`?

A. `14`

B. `10`

C. `18`

D. `15`

E. `13`

QUESTION 34

What is returned by `go(13)`?

A. `34`

B. `24`

C. `28`

D. `35`

E. `33`

```
public static int go( int x )
{
   if( x > 0 )
      return x + go(x-3);
   return x;
}
```

QUESTION 35

Which of the following correctly replaces `<*1>` in the code to the right such that `true` is printed?

A. `"kkkfuuekkshjjskk"`

B. `"kkekkksakkk"`

C. `"kdbvkkkkakkkek"`

D. `"hkakkkksliekkls"`

E. `"kkaodiikekkskkk"`

```
String x = <*1>;
out.print(x.matches("k{2}.+k{3}.+k"));
```

QUESTION 36

Simplify the Boolean algebra statement on the right.

A. true

B. `A(B + C)`

C. `A(!B + B) + A(!C + C)`

D. `!B + !C + A(B + C)`

E. false

```
!(BC)+A(B+!A)+AC
```

QUESTION 37

Which digital electronic symbol is shown at right?

QUESTION 38

What is the runtime for adding a new item to a Java Stack using the push() method?

QUESTION 39

What is the worst-case runtime to find an item in an un-balanced binary search tree?

QUESTION 40

Which data structure is typically used to create a breadth first search?

Name _____

School _____

Division: 1A 2A 3A 4A 5A 6A Grade: 9 10 11 12

Final Score _____ _____

2nd Grading _____ _____

1st Grading _____ _____
score initials

The Virtual Challenge Meets
Computer Science - Student Answer Sheet

1) _____
2) _____
3) _____
4) _____
5) _____
6) _____
7) _____
8) _____
9) _____
10) _____

11) _____
12) _____
13) _____
14) _____
15) _____
16) _____
17) _____
18) _____
19) _____
20) _____

21) _____
22) _____
23) _____
24) _____
25) _____
26) _____
27) _____
28) _____
29) _____
30) _____

31) _____
32) _____
33) _____
34) _____
35) _____
36) _____
37) _____
38) _____
39) _____
40) _____

Total Number Correct _____ x 6 = _____

Total Number Incorrect _____ x -2 = _____

Final Score (record here and top right) _____

Final scores can only be <u>even numbers</u> based on the above.
Odd-numbered scores will have to be DQed.