WASHINGTON STATE MIDDLE SCHOOL COMPUTER SCIENCE COMPETITION 2017

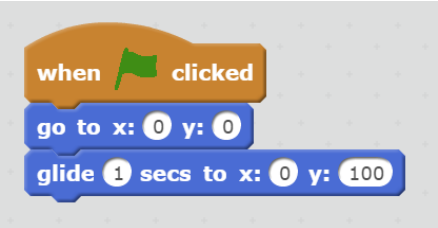
Individual Challenge
Grades 7 – 8
30 minutes

Please read these directions carefully before beginning. Breaking any of the rules is grounds for disqualification.

- Do not turn this page and begin working the test until the start of the test is announced. Once time starts, you will have **30 minutes** to complete this test.
- There is no talking allowed at any time. If you have questions, please raise your hand.
- No electronic devices of any kind (phones, computers, etc.) are allowed during the test.
- All questions are multiple choice questions. Mark all answers on the answer sheet provided. Mark as clearly as possible. The answer sheet is graded by an automated machine. If the machine is not able to read your answers, you will not receive points.
- Every question is worth 1 point. Some questions are more difficult than others. If you perceive a question to be difficult or is taking longer to solve, we recommend that you move on to the next question, and come back to it if you have time. Questions within each section approximately increase in difficulty.
- For all Scratch questions, assume that the script shown is the only script in the game.
- Do not feel bad if you don't finish the test - it is designed to be too long to finish in 30 minutes!

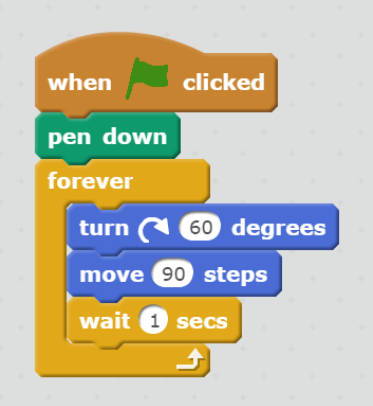
Scratch Questions

Question 1: What happens when the green flag is clicked?



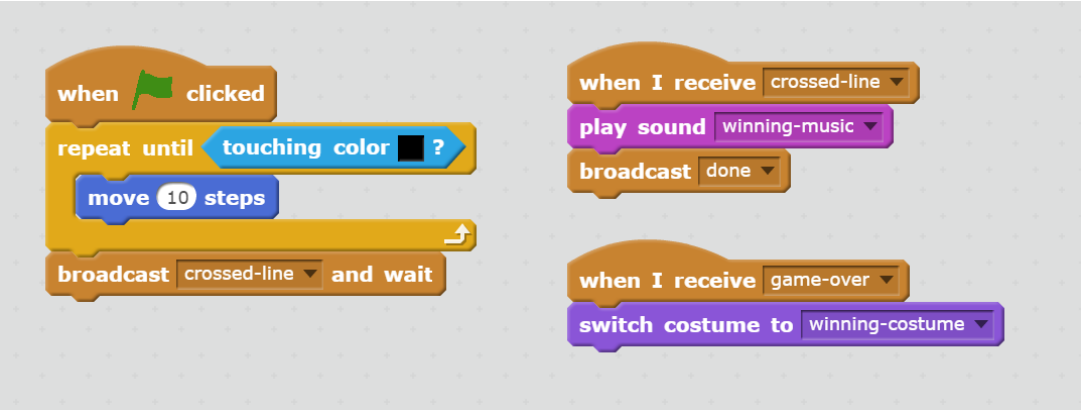
(A)	Nothing
(B)	Sprite glides to the middle of the screen
(C)	Sprite starts at the middle of the screen and glides downwards
(D)	Sprite starts at the middle of the screen and glides upwards

Question 2: When the green flag is clicked, what shape does the sprite draw?



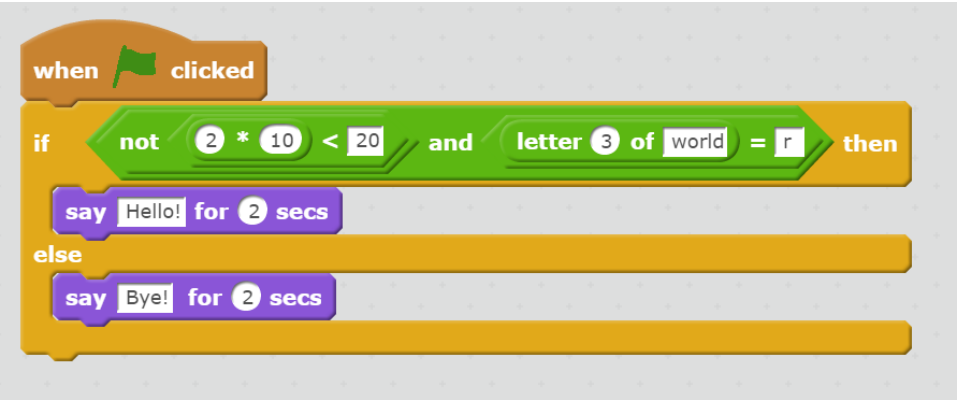
(A)	Octogen
(B)	Pentagon
(C)	Hexagon
(D)	Square

Question 3: What happens when the green flag is clicked?



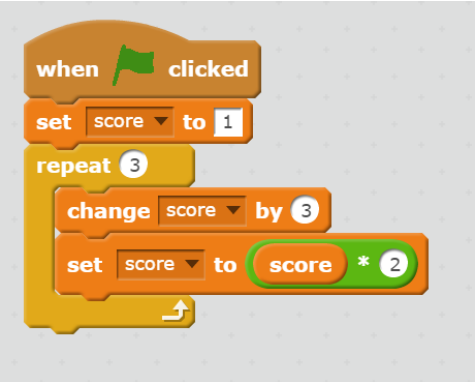
(A)	Sprite moves until it touches a black line and plays winning music
(B)	Sprite moves until it touches a black line and switches to winning costume
(C)	Sprite moves until it touches a black line, plays winning music and switches to winning costume
(D)	Sprite doesn't move

Question 4: What happens when the green flag is clicked?



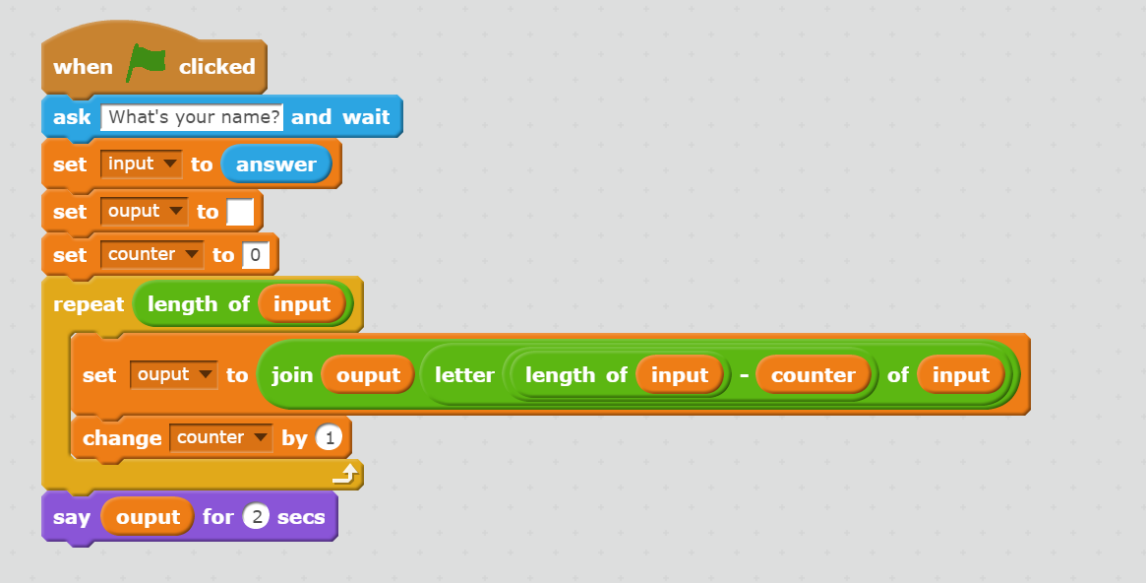
(A)	Nothing
(B)	Sprite says Hello!
(C)	Sprite says Bye!
(D)	Sprite says both Hello! And Bye!

Question 5: What value is stored in **score** after the code runs?



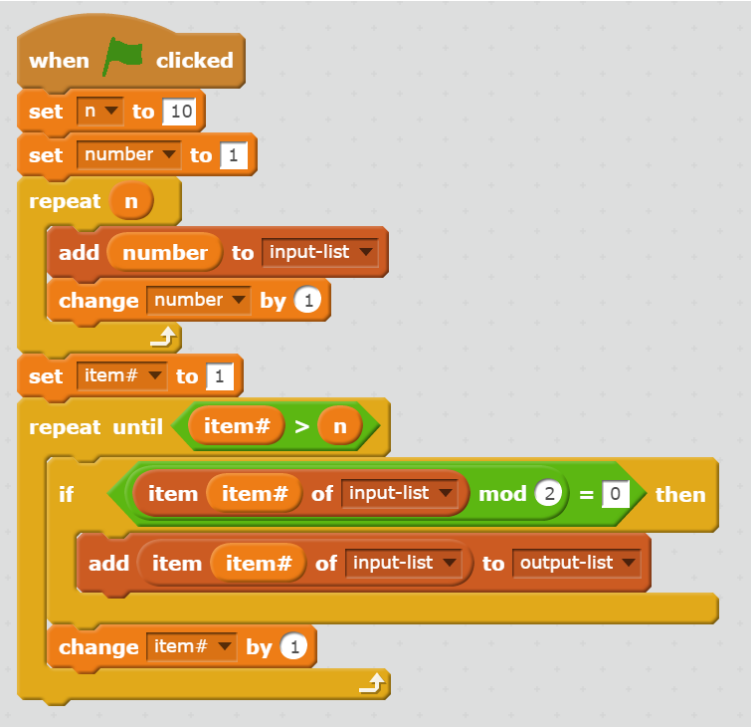
(A)	42
(B)	50
(C)	22
(D)	18

Question 6: If the input is typed as "How are you", what does the sprite say?



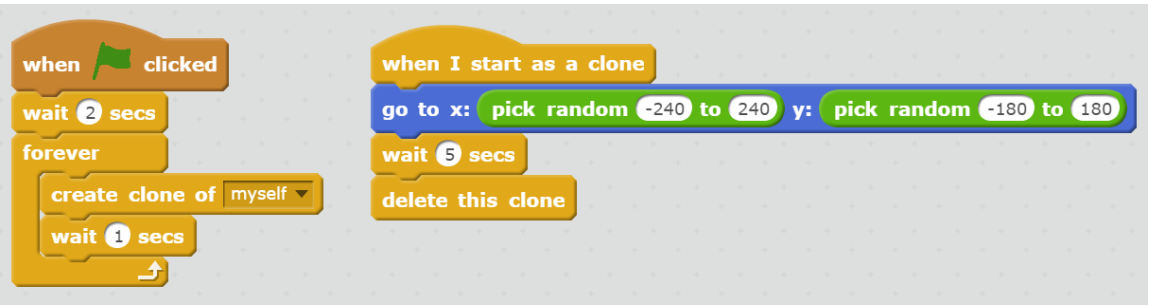
(A)	How are you
(B)	you are How
(C)	woH era uoy
(D)	uoy era woH

Question 7: When the following code finishes, what will be on the **input-list** and **output-list**?



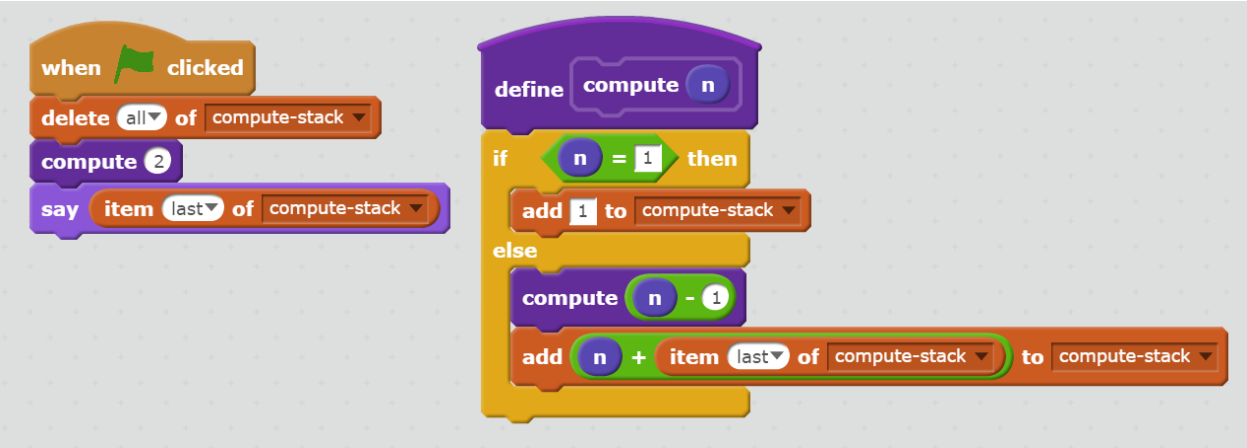
(A)	**input-list** will have 1 to 10 and **output-list** will have 2, 4, 6, 8, 10
(B)	**input-list** will have 1 to 10 and **output-list** will have 1, 3, 5, 7, 19
(C)	**input-list** will have 1, 3, 5, 7, 9 and **output-list** will have 2, 4, 6, 8, 10
(D)	**input-list** will have 2, 4, 6, 8, 10 and **output-list** will have 1, 3, 5, 7, 19

Question 8: How many clones will exist after 10 seconds?



(A)	1
(B)	5
(C)	6
(D)	8

Question 9: In the following program, the code in the **compute** block calls itself. What is this programming technique called?



(A)	Overloading
(B)	Stacking
(C)	Recursion
(D)	Overriding

Question 10: When the code above (in Question 9) finishes, what does the sprite say?

(A)	1
(B)	3
(C)	2
(D)	15

Pseudocode Questions

Pseudocode is a detailed yet readable description of what a computer program must do, expressed in a formally-styled natural language rather than in a programming language.

Question 11: It is **February** and a **Sunday**. Based on the following pseudocode, what is Ritu doing?

```
if (month is February) AND (day is Tuesday):
    Ritu is travelling
else if (month is June) OR (day is Sunday):
    Ritu is at home
else:
    Ritu is at work
```

(A)	Ritu is travelling
(B)	Ritu is at home
(C)	Ritu is at work
(D)	Ritu is travelling for work

Question 12: Ellen is a professor, and she grades her students based on their test scores using the following criteria:

Test Score	Grade
92 or above	A
80 to 91	B
79 or below	C

Which pseudocode will correctly assign the grades?

Pseudocode 1

```
if score > 91:
    grade = 'A'
else if score > 79:
    grade = 'B'
else:
    grade = 'C'
```

Pseudocode 2

```
grade = 'C'
if score >= 92:
    grade = 'A'
else if score >= 80:
    grade = 'B'
```

Pseudocode 3

```
if score >= 92:
    grade = 'A'
if score >= 80:
    grade = 'B'
if score <= 79:
    grade = 'C'
```

(A)	Pseudocode 1
(B)	Pseudocode 2
(C)	Pseudocode 1 or 2
(D)	Pseudocode 1, 2, or 3

Question 13: What does the following pseudocode print?

```
count = 5
repeat 3 times
    repeat count times:
        print 'Q'
    count = count - 1
    print line break
```

(A)	QQQQQ QQQQ QQQ
(B)	QQQQQ QQQQQ QQQQQ
(C)	Q QQ QQQ
(D)	QQQ QQ Q

Question 14: In the following pseudocode, if *input* is 11, what does it print?

```
N = input
while N > 0:
    if N is an odd number:
        print "1"
        N = (N – 1) / 2
    else:
        print "0"
        N = N / 2
```

(A)	1101
(B)	1011
(C)	0010
(D)	0100

Code Jumble

Consider the following 3 lines of pseudocode, each altering the value of **N**.

```
Line 1:  N = N x 2                    Multiply N by 2
Line 2:  if N > 4:  N = N – 10        subtract 10 only if N is greater than 4
Line 3:  N = N + 5                    add 5 to N
```

The value of **N is set to 0** before the pseudocode executes. (N = 0)
After the 3 lines execute, the value of N will be 5 based on the following calculation.

Line 1: 0 * 2 = 0
Line 2: 0 is not greater than 4, so do nothing
Line 3: 0 + 5 = 5

Suppose you can rearrange the 3 lines of code in any way you want. You will keep the same three lines of code, but you can reorder them. For example, you can move the second line to the top or you can move the first line to the bottom, etc.

N is set to 0 before execution.

Question 15: Find an arrangement (reordering of the 3 lines above) that will yield the maximum value for N. What is the value of N after it executes?

(A)	5
(B)	10
(C)	15
(D)	20

Question 16: Find an arrangement that will yield the minimum value for N. What is the value of N after it executes? Remember: -2 is less than -1. Hence, the minimum between the two numbers is -2.

(A)	0
(B)	-5
(C)	-10
(D)	-15

Functions

A **function** takes some value(s) as **input** and **returns** a value as **output**. We can write this process as ***FunctionName****(input) = output*. For example, if a function adds 2 to every input, we would have *AddTwoFunction(5) = 7*.

Shelly has a function called *MysteryNumberFunction*, but she cannot remember what it does. She tries 0, 1, 2, 8, and -2 as inputs and gets the following outputs:

```
MysteryNumberFunction(0) = 0
MysteryNumberFunction(1) = 0
MysteryNumberFunction(2) = 2
MysteryNumberFunction(8) = 56
MysteryNumberFunction(7) = 42
MysteryNumberFunction(-2) = 6
```

Question 17: What will Shelly see if she tries *MysteryNumberFunction*(10)?

(A)	100
(B)	121
(C)	72
(D)	90

Question 18: Shelly gets 12 as an output from *MysteryNumberFunction*. What could have been the input?

(A)	-4
(B)	3
(C)	either -4 or 3
(D)	either 4 or -3

Victor has another function called *MysteryStringFunction* that rearranges an input word in a certain order, but he also cannot remember what exactly it does. He tries “cat”, “victor”, “zyx”, “abcd”, “egf” as inputs and gets the following outputs:

```
MysteryStringFunction(“cat”) = “act”
MysteryStringFunction(“victor”) = “ciortv”
MysteryStringFunction(“zyx”) = “xyz”
MysteryStringFunction(“abcd”) = “abcd”
MysteryStringFunction(“egf”) = “efg”
```

Question 19: What will Victor see if he tries *MysteryStringFunction*(“lion”)?

(A)	“oiln”
(B)	“lion”
(C)	“ilno”
(D)	“ilon”

Question 20: Victor gets “opt” as an output from *MysteryStringFunction*. What could have been the input?

(A)	“pot”
(B)	“top”
(C)	“opt”
(D)	Any of the above

Arrays

An **array** is a data structure which stores items, or **values**, in a certain order. To create an array called ***ar*** containing numbers 12, 7, and 23, we would write the following:

```
Array ar = [12, 7, 23]
```

We can find a value by using an **array expression**, which includes the name of the array and the value's position to get that value back. Position starts at 1 and ends at the size of the array (3 in the above array). For instance:

```
ar[1] ---> 12
ar[2] ---> 7
ar[3] ---> 23
```

Consider the following array

```
Array arr = [1, 6, 5, 3, 4, 2]
```

Question 21: What is the value of `arr[3]`?

(A)	3
(B)	1
(C)	4
(D)	5

Question 22: What is the value of `arr[arr[3]]`? Note that the value of the inner array expression is used as the position for the outer array expression.

(A)	5
(B)	3
(C)	4
(D)	2

Question 23: Is there a position **i** and value **i** for which `arr[arr[i]] ---> i`? If so, what is the value of i?

(A)	Either 1 or 6
(B)	Either 6 or 5
(C)	Either 1 or 5
(D)	Either 2 or 4

Sorting

You are given an array of numbers 1 to 4 in a certain order.

```
Input: arr = [2, 1, 4, 3]
```

Your goal is to **sort** the array i.e., rearrange the numbers so that you get the following

```
Output: arr = [1, 2, 3, 4]
```

To sort them, the only operation you can apply is **Swap(i, j)**, which interchanges the number in position i (arr[i]) with the number in position j (arr[j]). For example, Swap(1, 2) on the above input will interchange arr[1] and arr[2] to give you the following result:

```
[1, 2, 4, 3]
```

If you further apply a Swap(3, 4) operation, you will get the desired end output.

```
[1, 2, 3, 4]
```

For the given input, we needed **two** Swap operations to sort the array.

Question 24: Suppose you have the following input:

```
Input: arr = [3, 4, 2, 1]
```

What is the minimum number of **Swap** operations you need to sort the input?

(A)	1
(B)	2
(C)	3
(D)	4

Question 25: Suppose the input is **reverse-sorted**, i.e., sorted in descending order.

```
Input: arr = [4, 3, 2, 1]
```

What is the minimum number of **Swap** operations you will need to sort the input?

(A)	1
(B)	2
(C)	3
(D)	4

Question 26: Instead of 4 numbers, suppose you have **n** numbers (from 1 to n) that are **reverse-sorted**. What is the minimum number of **Swap** operations you need to sort them?

(A)	**n – 1**
(B)	**n / 2** if n is even and **(n – 1) / 2** if n is odd
(C)	**n / 2** if n is even and **(n + 1) / 2** if n is odd
(D)	**n**

Logic and Math Questions

Question 27: Which conclusion follows from the statements below?

- `None of the hikers is an architect.`
- `All the campers are hikers.`

(A)	All hikers are architects
(B)	Architects are not campers
(C)	No hikers are campers
(D)	Some campers are architects

Question 28: Amy has more than two animals at home. All of them are dogs, except for two. All of them are cats, except for two. All of them are hamsters, except for two. How many animals does Amy have?

(A)	3
(B)	4
(C)	5
(D)	6

Question 29: You are given a standard deck of 52 cards (13 each of clubs, diamonds, hearts, and spades). When you pick a single card at random from the deck, what is the probability that you will pick a **King *or*** a **diamond** card?

(A)	1/52
(B)	1/4
(C)	16/52
(D)	17/52

Question 30: What is the next number in the following sequence?

`1, 8, 27, 64, __`

(A)	100
(B)	225
(C)	125
(D)	625

Question 31: In 2 years, Mark will be 3 times Eva's age. In 8 years, Eva will be half of Mark's age. What is Mark's age today?

(A)	16
(B)	18
(C)	20
(D)	22

Question 32: A car is going up a hill. The hill is one mile long. The driver goes up the hill at an average speed of 30 miles per hour. When the driver reaches the top of the hill, he starts down the other side. The downhill side is also one mile long. How fast must the driver go down the hill in order to average 60 miles per hour for the entire trip?

(A)	210 miles per hour
(B)	90 miles per hour
(C)	60 miles per hour
(D)	It is impossible to average 60 miles per hour

Question 33: Twelve teams are participating in a soccer tournament. The teams are split into 2 groups with 6 teams each. In the first round, **inside each group**, **every team plays against every other team *twice*** (once each in their home ground). How many total games are played in the first round?

(A)	36
(B)	30
(C)	60
(D)	72

Question 34: An intelligent trader is driving from one place to another carrying 3 sacks having 30 potatoes each. No sack can hold more than 30 potatoes. On the way, she passes through 30 checkpoints and on each checkpoint, she has to pay a tax of 1 potato per sack she is carrying. What is the maximum number of potatoes she can carry past the checkpoints?

(A)	25
(B)	30
(C)	0
(D)	15

Bonus Questions

Searching

The following pseudocode searches for a number **e** in an array **arr** with **n** numbers. The code prints "found" and *exits* when the number is found. If the number is not present, it prints "not found". **count** keeps track of the number of iterations done before the number is found. As with previous array examples, the array positions start at 1 and end at n.

```
arr = input array
n = size of the input array
e = input number to search
count = 0
for i = 1 to n
    if arr[i] is equal to e
        print "found"
        exit
    else
        count = count + 1
print "not found"
```

For the example input `arr = [11, 7, 23, 8, 19, 3, 1], n = 6, and e = 19`, the output will be "found". The value of *count* will be **4** before the code exits.

Question 35: If the number to be found is in the last position of the array (at arr[n]), what will be the value of **count** when the pseudocode exits?

(A)	0
(B)	n
(C)	n – 1
(D)	n – 2

Question 36: If the number to be found is not present in the array, what will be the value of **count** when the pseudocode exits?

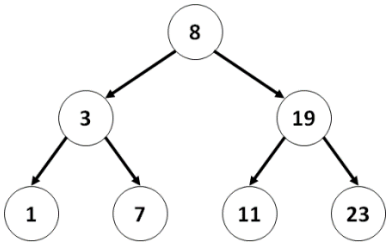
(A)	n + 1
(B)	n
(C)	n – 1
(D)	n – 2

Binary Tree

The pseudocode to search for a number described in the previous question will scan the entire array in the worst-case scenario. This can be slow when the set of numbers is large. Programmers often use a data structure called **binary tree** to speed up the search.

For the following set of numbers,
`[11, 7, 23, 8, 19, 3, 1]`

The binary tree structure looks like the following



Binary tree has a set of **nodes**. Each node holds one of the numbers as its **value**. The tree has a **root node**. In the above tree, 8 is the root node. Each node is attached to a *left child node* and a *right child node* except for the **leaf nodes**. 3 is the *left child node* of 8 and 19 is the *right child node* of 8. The leaf nodes are 1, 7, 11, and 23. The numbers are organized in a way that the following pseudocode can be used to search if a number is present.

```
x = root of the tree
e = input element to search
count = 0
repeat until exit:
    if value at x is equal to e:
        print “found”
        exit
    else
        count = count + 1
        if x is a leaf node:
            print “not found”
            exit
        if e is less than value at x:
            set x as left node of x
        else if e is greater than value at x:
            set x as right node of x
```

count keeps track of the number of iterations done before the number is found.

Question 37: How are the nodes in the binary tree organized?

(A)	nodes are organized at random
(B)	the value in a node is less than the value in the left child node the value in a node is greater than the value in the right child node
(C)	the value in the root node is great than values in leaf nodes
(D)	the value in a node is greater than the value in the left child node the value in a node is less than the value in the right child node

Question 38: Suppose we are searching for the number 11 in the above binary tree with 7 nodes. What is the value of **count** before 11 is found and the pseudocode exits.

(A)	0
(B)	1
(C)	2
(D)	3

Question 39: If the number to be found is not present in the binary tree, what will be the value of **count** when the pseudocode exits?

(A)	1
(B)	2
(C)	3
(D)	4