CAMPEAU TEST
MULTIPLE CHOICE QUESTIONS
VERSION 0.85

PART A – NO OOP AND NO STRINGS NO ARRAYLISTS

This is a simple Multiple Choice test to prepare you for the Multiple Choice Questions on the Computer Science AP exam.

Please note that many of these questions are easier than the ones on the AP exam in order to help you remember different concepts.

Q-What will be printed by the following code?

```java
public class CampeauTest
{
     public static void main(String[] args)
     {
          int i = 11;
          int j = 2;
          double k = 4.0;
          System.out.println(k + i * j);
     }
}
```

A) 8.022
B) 26.0
C) 30.0
D) 125
E) 1515

SOLUTION

Answer B) 26.0

EXPLANATION

The key is to remember that bedmas applies.
Topics: BEDMAS

Q-What will be printed by the following code?

```java
public class CampeauTest
{
     public static void main(String[] args)
     {
          int i = 9;
          int j = 2;
          double k = 4.0;
          System.out.println(2 + i / j * k - 1);
     }
}
```

A) 16.0
B) 17.0
C) 19.0
D) 21.0
E) 25.0

SOLUTION

Answer B) 17.0

EXPLANATION

The key is to remember that bedmas applies.
Topics: BEDMAS, INTEGER DIVISION

Q-What will be printed by the following code?

```
public class CampeauTest
{
     public static void main(String[] args)
     {
          int a = 3;
          int b = 2;
          double c = 7.0;
          System.out.println(c * a / b);
     }
}
```

A) 6.0
B) 7.0
C) 10.0
D) 10.5
E) 13.0

SOLUTION

Answer D) 10.5

EXPLANATION

The key is to do the c * a first and remember that the result is a double. So then the division is a normal division instead of a integer division.
Topics: BEDMAS, INTEGER DIVISION, OPERATIONS BETWEEN DOUBLE AND INT

Q-What will be printed by the following code?

```
public class CampeauTest
{
     public static void main(String[] args)
     {
          int a = 18;
          int b = 5;
          System.out.println(22 % 5);
     }
}
```

A) 1
B) 2
C) 3
D) 4
E) 4.4

SOLUTION

Answer B) 2

EXPLANATION

You simply need to remember that % means mod and that 22 mod 5 is 2 because 22 divided by 5 is 4 remainder 2.
Topics: MOD

Q-Which number will be printed by the following code?

```
public class CampeauTest
{
    public static void main(String[] args)
    {
        for (int i = 18; i < 50; i = i + 7)
            System.out.println(i % 5);
    }
}
```

A) 8, 5, 2, 9, 6
B) 3, 5, 2, 4, 1
C) 3, 5, 2, 4, 6
D) 3, 0, 2, 4, 6
E) 3, 0, 2, 4, 1

SOLUTION

Answer E) 3, 0, 2, 4, 1

EXPLANATION

First, we need to realized that we have a one-statement for-loop, so no brackets are required.

Second, we need to figure out that the numbers that the control variable i will get are 18,25,32,39,46 for the different iterations.

Now, we just do the following calculations:

18 % 5 = 3
25 % 5 = 0
32 % 5 = 2
39 % 5 = 4
46 % 5 = 1

Topics: MOD, for loops

Q-Which of the following statements assigns a random integer between 0 and 5, inclusively, to the variable rn?

```
A) int rn = (int) (Math.random() * 5) + 1;
B) int rn = (int) (Math.random() * 5) + 0;
C) int rn = (int) (Math.random() * 4) + 1;
D) int rn = (int) (Math.random() * 6) + 0;
E) int rn = (int) (Math.random() * 1) + 4;
```

SOLUTION

Answer `D) int rn = (int) (Math.random() * 6) + 0;`

EXPLANATION

You need to remember that Math.random() gives a number n that is greater or equal to zero and less or equal to 0.999999999… (so less than 1). So, Math.random() * 6 gives us a random number from 0.0 to 5.9999999… When we take the integer of this, we get a range of 0 to 5.
Topics: random numbers, type casting

Q-Which of the following statements assigns a random integer between 14 and 25, inclusively, to the variable rn?

```
A) int rn = (int) (Math.random() * 12) + 14;
B) int rn = (int) (Math.random() * 11) + 14;
C) int rn = (int) (Math.random() * 14) + 11;
D) int rn = (int) (Math.random() * 25) - 12;
E) int rn = (int) (Math.random() * 14) + 12;
```

SOLUTION

Answer `A) int rn = (int) (Math.random() * 12) + 14;`

EXPLANATION

The range produced by (int) Math.random() x 12 is 0 to 11. By adding 14 to each, we get the range 14 to 25.

Topics: random numbers, type casting

Q-Consider the following method named letterMark. It was implemented incorrectly by a bad programmer. For which value (out of the ones listed) of grade will this method return the correct letter?

```
public static String letterMark(int grade)
{
    if (grade >= 80)
        return "A";
    else if (grade <= 70)
        return "B";
    else if (grade >= 60)
        return "C";
    else if (grade <= 50)
        return "D";
    else
        return "F";
}
```

A) 45
B) 55
C) 65
D) 75
E) 85

SOLUTION

Answer E) 85

EXPLANATION

There are two errors in the code. Both lesser than symbols < should be replaced by greater than symbols >.

Any mark equal to 80 or above will lead to an A which is correct.

Any mark below 70 will lead to a B which is incorrect.

This leaves only marks between 70 and 80. These will actually be given a C which is also incorrect.

So, only As work correctly in the current version of the code.

Topics: If statements, Conditional operators

Q-What will the following code output to screen?

```
public class CampeauTest
{
     public static void main(String[] args)
     {
          boolean a = true;
          boolean b = false;
          System.out.println(a && b);
     }
}
```

A) true && false
B) true
C) false

SOLUTION

Answer C) false

EXPLANATION

The key here is to remember that && means AND.

We also need to remember that *true AND false* gives false because the AND operator only gives a true result when both sides are true.

Topics: && operator

Q-What will the following code output to screen?

```
public class CampeauTest
{
     public static void main(String[] args)
     {
          boolean a = true;
          boolean b = false;
          System.out.println(a || b);
     }
}
```

A) true && false
B) true
C) false

SOLUTION

Answer B) true

EXPLANATION

The key here is to remember that || means OR.

We also need to remember that OR gives a true result as long as at least one of the two values is true. So, true or false gives us true.

Topics: || operator

Q-What will the following code output to screen?

```
public class CampeauTest
{
     public static void main(String[] args)
     {
          boolean a = true;
          boolean b = false;
          boolean c = !a || b;
          System.out.println(c);
     }
}
```

A) not true
B) not false
C) true
D) false

SOLUTION

Answer D) false

EXPLANATION

We have to know what the ! symbol does. It stands for NOT.

The value of !a is not true which is false.

So, c gets the value of false OR false which is false.

The program then outputs false.

Topics: ! operator

Q-What will the following code output to screen?

```
public class CampeauTest
{
     public static void main(String[] args)
     {
          boolean a = true;
          boolean b = false;
          boolean c = false;
          System.out.println(a || b && c);
     }
}
```

A) true
B) false

SOLUTION

Answer A) true

EXPLANATION

If you got this wrong, you probably forgot that the && operator is evaluated before the || operator.

Topics: && and || operator precedence

Q-What will the following function return?

```
    public static boolean merger(boolean x, boolean y)
    {
        return x || (y && !y);
    }
```

A) It will always return true.
B) It will always return false.
C) It will return whatever the value of x is.
D) It will return whatever the value of y is.
E) It will return the opposite of y.

SOLUTION

Answer C) It will return whatever the value of x is.

EXPLANATION

We need to analyze the Boolean expression x || (y && !y).

We first look at what is in brackets. Notice that no matter what y is, the value inside the brackets will be false.

So, this simplifies the expression to x || false. And this is the same as just x.

Topics: Analyzing Boolean expressions.

Q-What will the following code output to screen?

```
public class CampeauTest
{
	public static void main(String[] args)
	{
		int a = 10;
		while (a > 2)
		{
			System.out.print(a + " ");
			a -= 3;
		}
	}
}
```

A) 10 3
B) 10 7 4
C) 10 8 6 4
D) 10 8 6 4 2
E) 10 7 4 1

SOLUTION

Answer B) 10 7 4

EXPLANATION

You need to remember that the -= symbol means to subtract from.

So, the line `a -= 3;` could have been written as `a = a - 3;` instead.

If you are unsure about this, you should also look up +=, *= and /= before the test.

Topics: Arithmetic operators

Q-What will the following code output to screen?

```
public class CampeauTest
{
     public static void main(String[] args)
     {
          int val = 24;
          int div = 3;
          while ((val % 2 == 0))
          {
               if (val % div == 0)
               {
                    System.out.print(val + " ");
               }
               val /= 2;
          }
     }
}
```

A) 24 22 20 18 16 14 12 10 8 6 4 2 0
B) 24 12 6 3
C) 24 12 6
D) 24 12 6 3 0
E) Nothing will be outputted.

SOLUTION

Answer C) 24 12 6

EXPLANATION

This is messy. And the code is not optimized. Notice that the div variable never changes. So we could simply replace div by 3 everywhere.

We loop as long as val is even. And in each loop, val halved. So we get the values 24 12 6 and 3 (but we do not loop for 3 as it is not even).

So, for the passes where val is 24, 12 and 6, we output them to screen if they are divisible by three. They all are, so we output them all to screen.

Topics: Loops, mod, the /= operator

Q-Consider the four segments of code below. Which one(s) will output the content of the array data to screen?

```
I)

System.out.println(Arrays.toString(data));
```

```
II)

for(int i=0; i<data.length; i++)
{
    System.out.print(i + ", ");
}
```

```
III)

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

```
IV)

for(int i=0; i<data.length; i++)
{
    System.out.print(data[i] + ", ");
}
```

A) I, II, III and IV
B) I, III & IV
C) II, III & IV
D) I & IV
E) II & III

SOLUTION

Answer D) I & IV

EXPLANATION

The code segment II outputs the index number instead of the element value. The code segment III outputs the reference to the array.

Segments I and IV both output the content of the array.

Topics: Arrays, Outputting arrays

Q-Consider the function *remake* shown below. If each array in the possible solutions was passed to the function, which array would then have its last element be the value 19?

```
public static void remake(int[] arr)
{
    for (int i=2; i<arr.length; i++)
    {
        arr[i] = arr[i-1] + arr[i-2];
    }
}
```

A) {18, 1, 1, 7}
B) {3, 4, 1, 7, 1}
C) {2, 5, 8, 0, 1}
D) {1, 4, 1}
E) {3, 2, 3, 16, 5}

SOLUTION

Answer C) {2, 5, 8, 0, 1}

EXPLANATION
First, you need to remember be comfortable with the idea that you can create an array using { and }.

Second, you have to remember that arrays are passed by reference which means that they are changed in the method (function) then they will be changed in the original location as well.

After those to steps, you are ready to analyze the for loop. Notice that each element I in the loop is set to the two elements for it. And notice that i begins at 2. So, arr[2] is set to arr[0] + arr[1], then arr[3] is set to arr[2] + arr[1], and so on…

So, when looking at the possible array solutions, we only need to look at the first two element values. Of course, we also need to consider the length of the array.

For the array in C, we get the new array: {2, 5, 7, 12, 19}

Topics: Arrays, Passing arrays to methods, Manipulating arrays

Q-What will the following code output to screen?

```
public class CampeauTest
{
     public static void main(String[] args)
     {
          int[] arr = {5, 3, 9, 1};
          int total = 0;
          for (int z: arr)
          {
               total += z;
          }
          System.out.println(total);
     }
}
```

A) 5391
B) 5
C) 18
D) 1
E) 72

SOLUTION

Answer C) 18

EXPLANATION

The key here is remembering how the enhanced for loop works.

In each iteration of the loop, z takes on a different element value from array arr.

In the end, total is the total of all values in the array.

Topics: Arrays, Enhanced for loop, += operator

Q-Consider the method called mystery below. What does it do?

```
public static int mystery(int[] arr)
{
     int count = 0;
     for (int v: arr)
     {
          if (v % 2 == 0)
          {
               count++;
          }
     }
     return count;
}
```

A) It counts the number of elements in the array.
B) It totals the value of the elements in the array.
C) It totals half of the elements in the array.
D) It counts the number of even numbers in the array.
E) It counts the number of occurrences of the value 2 in the array.

SOLUTION

Answer D) It counts the number of even numbers in the array.

EXPLANATION

The key is to remembering how the enhanced for loop works. After that, you need to recall that v % 2 == 0 will be true whenever v is even. After that, you should be able to understand the answer.

Topics: Arrays, Enhanced for loop, MOD

Q-What will be outputted to screen?

```
public class CampeauTest
{
    public static void main(String[] args)
    {

        String[][] arr = {{"A", "B", "C"},
                          {"D", "E", "F"},
                          {"G", "H", "I"}};
        System.out.println(arr[1][2]);
    }
}
```

A) A
B) B
C) C
D) D
E) E
F) F
G) G
H) H
I) I

SOLUTION

Answer F) F

EXPLANATION
We need to remember that {"A", "B", "C"} makes up row 0, {"D", "E", "F"} makes up row 1 and {"G", "H", "I"}}; makes up row 2. And, we need to remember that ADG are in column 0, BEH in column 1 and CFI are in column 2.

Finally, we have to remember that we use [row][column] for indexes in 2D arrays.

So, the values are row 1 and column 2 is F.

Topics: 2D arrays

Q-What will be outputted to screen?

```
public class CampeauTest
{
     public static void main(String[] args)
     {

          int[][] arr = {{1, 2, 3},
                         {4, 5},
                         {6, 7, 8}};
          int sum = 0;
          for (int[] x : arr)
          {
               for (int i = 0; i < x.length; i++)
               {
                    sum = sum + x[i];
               }
          }
          System.out.println(sum);
     }
}
```

A) sum
B) 6
C) 9
D) 23
E) 36

SOLUTION

Answer E) 36

EXPLANATION
The enhanced for loop with a 2D array is a little tricky. Because a 2D array is really an array of arrays, each element from an enhanced for loop ends up being an array. So x is in fact an array.

Once you have that figured out, you can see how the inner loop iterates over the array totaling it up. After every pass, every element in the array is totaled up.

So, 1 + 2 + 3 + 4+ 5 + 6 +7 + 8 gives 36.

Topics: 2D arrays, Enhanced for loop

Q-Consider the method below. It will cause an error in some situations. What needs to be changed to stop the error from happening?

```
public static boolean contains(int[][] arr, int value)
{
     for (int r=0; r<arr.length; r++)
     {
          for (int c=0; c<arr[0].length; c++)
          {
               if (arr[r][c] == value)
               {
                    return true;
               }
          }
     }
     return false;
}
```

A) Change the `arr[0].length` to `arr[r].length`.
B) Change the `return true` to `return false`, and the `return false` to `return true`.
C) Change `c < arr[0].length` to `c <arr[0].length - 1`
D) Change `arr[r][c]` to `arr[c][r]`.
E) Make the outter for loop loop from `arr.length-1` down to `0` instead of `0` up to `arr.length-1`.

SOLUTION

Answer A) Change the `arr[0].length` to `arr[r].length`.

EXPLANATION

The key here is to remember that 2D arrays are not necessarily rectangular. The function above will give an index-out-of-bounds error if one of the rows has less elements than row 0. Furthermore, if one of the rows has more elements than row 0, we would actually skip those elements causing incorrect results.

To fix this problem, we change the inner loop so that it loops for the amount of elements in the current row.

Topics: 2D arrays

Q-The static method named `area` gets two `double` parameters that represent the `width` and `height` of a rectangular shape and calculates the area of that rectangle and returns it. Which of the following could be a correct method header (prototype) for this function?

```
I)
public static double area(double w, double h)

II)
public static void area(double w, double h)

III)
public static int area(int w, int h)

IV)
public static void area()

V)
public static double area()
```

A) I
B) II
C) III
D) IV
E) V

SOLUTION

Answer A) I

EXPLANATION

In the description, we are told that there are two double parameters. So that eliminates options III, IV and V. We are also told that the area will be returned. So the return-type cannot be void leaving us with only option I.

Topics: method headers

Q-What does the following method do?

```
public static int mystery(int n)
{
     if (n == 1)
     {
          return 1;
     }
     else
     {
          return n * mystery(n-1);
     }
}
```

A) It returns the value of n * n.
B) It returns the value of n factorial.
C) It outputs all numbers from n down to 0.
D) It returns n numbers.
E) It returns the sum of 1 + 2 + 3 + ... + n.

SOLUTION

Answer B) It returns the value of n factorial.

EXPLANATION

When we are unsure of what a method does, we should try a function call with a value that is close to the escape case.

Let's try mystery(3).

This will return 3 * mystery(2).

This then becomes 3 * 2 * mystery(1).

This then becomes 3 * 2 * 1 which is 6.

This is factorial. If unsure, you can try a call with other values to be certain.

Topics: recursion

To add:
-For loop problem
-Which of the following will correctly swap element values in an array
-Nested for loop problem

To add in Part B:
ArrayLists
Strings
Everything about OOP