Q1: Choose the best answer for each of the following.

1	**What does the keyword static mean in the declaration of the main method?**			
(A) The method is not visible outside of the class	(B) The method has no parameters	(C) The method has no return value	(D) The method can be called without creating an instance	
2	**How is the return value declared in a method?**			
(A) By the data type and the variable name following the method name	(B) By the data type alone before the method name	(C) By the data type and variable names in parentheses	(D) By the data type alone after the method name	
3	**Which of the following is NOT true about a method in Java?**			
(A) Java methods can have no parameters	(B) Java methods can return more than one value	(C) Java methods can return no values	(D) Java methods can have more than one parameter	
4	**Methods eliminate the need for duplicate statements.**			
(A) True	(B) False			
5	**A program statement or expression that transfers control to a method so that the method will perform its particular subtask is called a/an**			
(A) Parameter	(B) Argument	(C) Method declaration	(D) Method call	
6	**A value supplied to a method is known as a/an**			
(A) Parameter	(B) variable	(C) argument	(D) return value	
7	**When the type of a method is defined as void, this indicates**			
(A) The method returns zero	(B) The method returns nothing	(C) The method returns empty value	(D) The method returns null	
8	**In the Method heading below, a and b are called..** **public double sum (int a, char b)**			
(A) Parameters	(B) Arguments			

9	**In the following method declaration, what is the return type?** **public static int myMethod(int count, double value) { return 4; }**			
	(A) myMethod	(B) 4	(C) int	(D) double
10	**In the following method declaration, what is the name of the method?public** static void showMenu(String category) { }			
	(A) static	(B) category	(C) void	(D) showMenu
11	**In the following method declaration, how many formal parameters are there?** public static double squareRoot(double value) { return 2.3;}			
	(A) 0	(B) 1	(C) 2	(D) None; methods do not have formal parameters

12	**The following code in the main is valid**
	public static void main(String [] args) {
	int value; value = createGameCharacter();
	// rest of program
	}
	public static void createGameCharacter() {}			

	(A) True	(B) False		

13	**What is the value that is returned in the following method?**
	public static int getVolume() {
	int v;
	v = 23;
	if (v < 50) {
	return 50;
	}
	return v;
	}			

	(A) 50	(B) 23	(C) 100 and 50	(D) 100
14	Math is a predefined class which can be imported from the package.			
	(A) java.util	(B) java.lang	(C) java.com	(D) java.net

Q2: State which of the following is true / false

1	The part from the program that performs specific task is known as method	T
2	Divide and conquer it's a programming approach that aims to divide the program to small pieces and then combine these pieces to build the entire program.	T
3	Java Application Programming Interface(Java API) has a large collection of classes and methods that you can combine with your own methods to build programs.	T
4	One of the most advantages of using method is the ability of re-using the code.	T
5	Related classes are typically grouped into one File	F
6	We write method once and reuse it whenever we need to perform the same task	T
7	Use existing classes and methods as building blocks to create new programs known as program compilation.	F
8	It's a good approach to let the method perform many tasks.	F
9	Method with one task is easier to test and debug.	T
10	It's better to give each method a name that is related to the method task.	T
11	The method that we can create in Java is the main method only.	F
12	When we want to perform the method task we need to call it first.	T
13	We can call the method from main method only.	F
14	Other methods can call each other	T
15	Methods that don't depend on any object known as static methods.	T
16	Static method performs common task	T

17	To define a static method we use the keyword *public*	F
18	To call a static method we call it by class name and period(.) and then the method name	T
19	One of the most classes that contains common static methods is Math class.	T
20	Class System provides a collection of static methods that enable you to perform common mathematical calculations.	F
21	Class Math is a part of java.util package	F
22	We don't need to import the class Math when we want to use it because it's located in java.lang package so that the compiler will import it implicitly	T
23	tan(x) is a static method in Math class that find the square root the number x	F
24	In the program we can have local variables and instance variables.	T
25	Every object has its own copy of instance variables	T
26	Every object has its own copy of static variables	F
27	Static variables are common between all objects in the program	T
28	For each static variable only one copy will be created and it shared between all objects.	T
29	Class's fields are instance variables with static variables together	T
30	Static variables known as class variables	T
31	We declare constants in java with the keyword final	T
32	Constant in class Math declared with the public , static and final modifiers	T
33	Every method or variables declared with the keyword public we can use it in other classes.	T
34	If the variable declared as final then it means we can't change its value	T

35	One of the most important static variables declared with final keyword in class Math are PI and E	T
36	main method declared static to enable JVM to call the main method and run the program even if there's no instance(object) of the class.	T
37	Each method contains a header that contains access modifier such as (public) , method type ,method name with parenthesis containing method's parameters list	T
38	It's allowed to define a method without specifying the type or name of it	F
39	It's allowed to define a method without specifying a parameter list for it	
40	Parameter is a variable consists of data type and name only	T
41	We use parameters when we need to pass some values to the method to perform the task on them	T
42	Each method can have any number of parameters	T
43	We operate parameters by semicolon (;)	F
44	We can give the parameter defined in the method header a value like int x = 10 ;	F
45	The name of a parameter in the method declaration is called a formal argument or a formal parameter.	T
46	When we call the method we replace the parameters by actual variables	T
47	We can have two parameters with the same name	F
48	We can have two parameters with the same data type but different name	T
49	Number of parameters depends on the method task	T
50	Each method should have at least one parameter	F
51	We can consider parameters as input for the method	T
52	**Always** parameters list comes after **method name**	T

53	It's allowed to post parameters list inside square brackets [ ] instead of parenthesis ( )	F
54	Method are of two types , a method that returns a value and a method that does not return any vlaue	T
55	Method that return a value declared with the keyword void	F
56	If the method declaration contains the keyword means that method should not return any value	T
57	If the method will return a string value then it should declared String instead of void	T
58	If the method will return an integer value then it should declared int instead of void	T
59	If the method will return a value then we decide about the type we declare in the method header depending on the value that we want to return	T
60	We declare method void or return depending on the task that the method will do. If the method needs to return a value then we declare it as a returned value otherwise void to indicate that the method will not return any value	T
61	The value the method will return will be sent back to the caller method	T
62	Caller method is always the main method.	F
63	Caller method is the method that will call another method	T
64	Method declared void can have a return statement in its body to stop the method execution.	T
65	return keyword always written in the body of the method	T
66	We sue return statement to return the value or to stop the method execution if it's void	T
67	return key word inside a void method is **optional**	T
68	If a method declared returned type , then the return keyword existing in the body is **optional**	F
69	Method that declared returned value type then it **MUST(should)** return a value	T
70	Generally , we can have two return statements in the method's body	F

71	We can have two return statements in the method's body if we control them using if statement.	T
72	return statement should be last statement in the body	T
73	Not allowed to write any statement after return statement unless we use if statement	T
74	A method can return at many values	F
75	A method can return at most 1 value	T
76	It's a compilation error when you forget to return a value from a returned value method	T
77	When we call a method we should replace formal parameters by actual values	T
78	The Type, order and number of parameters is so important when we call the method	T
79	For a method declared with three parameters we can call it and pass only two parameters.	F
80	If we want to return many values from a method then we should use object reference concept	T
81	If we want to access a variable from many method then we should declare it as a field (inside the class but outside all methods)	T
82	Non-static methods are typically called instance methods.	T
83	We can call the non-static method by its class name	F
84	We can call method declared in the same class directly by its name	T
85	We can call a non-static method from a static method	F
87	To call a non-static method we need reference variable then we use it to call the method.	T
87	When the method finishes the control will return to the statement that made the call to the method	T
88	Declaring a method outside the class is a fatal logic error	F

89	Re-declaring parameters as local variable inside the method body is a compilation error	T
90	We use a data structure called a stack to manage (control) the methods call and returned addresses.	T
91	Stack frame is a part of method call stack that allocates a memory space for local parameters and parameters of the method being called	T
92	When we call a method , it pushed (added) to the stack and when the method finished and wants to return to the called we make a pop(remove) operation from the stack	T
93	If many methods occurred and the stack can't store them then a stack overflow error will occur	T
94	Java contains many predefined classes that are grouped into categories of related classes called packages.	T
95	java.lang package contains classes and interfaces required for each program and they are imported implicitly by the compiler	T

Q3:Find the syntax errors in each of the following code segments.

S.N	Code Segment

1	`public static multiply(){`
	`System.out.println(" 6x5 = "+5*5);`
	`}`

2	`public static sum void (){`
	`System.out.println(" 10+7 = "+(10+7));`
	`}`

3	`public static void main(String[] args) {`
	`int x = 10;`
	`}`
	`public void printX() {`
	`System.out.println("x= "+x);`
	`}`

4	`public static void main(String[] args) {`
	`System.out.println("welcome");`
	`displayName();`
	`System.out.println("Bye !");`
	`}`
	`public void displayName() {`
	`System.out.println("My Name is Naif.");`
	`}`

```
5       public static void method1(int a) {

            int a = 5;
            System.out.println(a);

        }
```

```
6       public static void method1(int var, String var) {

            System.out.println("My Name is: " + var);
            System.out.println("My Age is: " + var);

        }
```

```
7       public static void myMethod(int , double , int) {


        }
```

```
8       public static void method1( int a : int b) {
                System.out.println("a+b="(a+b));
                return ;

        }
```

```
9       public static void method1(int a) {

            return;
            System.out.println("a= " + a);

        }
```

```
10      public static void method1(int a) {

            if (a <= 5) {
                return;
            }
            else
                System.out.println("a is not less than or equal five");

            return ;
            System.out.println("a= " + a);

        }
```

```
11      public static int method1(String name) {

            System.out.println("My name is : " + name);

        }
```

```
12      public static int getValue(String name , int age) {

            System.out.println("My name is : " + name);
            System.out.println("My Age is : "+age);
            return name;
        }
```

```
13      public static int getValue() {

            int a;
            a = 5;

            double b = a;

            return b;
        }
```

```
14      public static int getValue()

            char a = 100;

            return a;
```

```
15      public static int getValue() {

            byte a = 10;

            short b = 20;

            return a , b;
        }
```

```
16      public static int getValue() {

            byte a;
            return a;
        }
```

```
17      public static double getValue {

            int a = 5;

            return a;
        }
```

```
18      public static int getValue() {

            double a = 5;

            return (double)a;
        }
```

```
19      public static char getValue() {

            System.out.println("welcome");
            return ;
        }
```

```
20      private static boolean isEven(int a) {

            if (a % 2 == 0) {
                return true;
            } else {
                return false;
            }

            return false;
        }
```

21
```
*   private static String getNumberString(int a) {
        switch (a) {
            case 1:
                return "One";
                break;
            case 2:
                return "Two";
                break;
        }
    }
```

2
```
    public String getNumberString(int a) {

        switch (a) {
            case 1:
                return "One";

            case 2:
                return "Two";

            default:
                return "Unknown";
        }
        return "Invalid";
    }
```

23
```
    public String myMethod(int a) {

        for (int i = 0; i < a; i++) {
            return "str";
        }
    }
```

24
```
   public static void square(int num) {
        return num * num;

    }
```

25
```
    public static void main(String[] args) {

        myMethod();
        myMethod("car");
        myMethod(10);
        myMethod(10 , "car");
        myMethod ('car',10);
        myMethod("car":10);
        myMethod("car",10.0);
        myMethod("car",(double)10);
        myMethod("car",(byte)10);
        myMethod("car",10);
        mymethod("car",10);


    }

    public static void myMethod(String s, int a) {

        /*
        method body goes here
        */
    }
```

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

            double num = 4;

            isEven(num);

        }

        public static boolean isEven(int num) {

            if (num % 2 == 0)
                return true;


            return false;

        }
```

Q4:Find the output for each of the following code segments.

S.N	Code Segment	Output

1	`public static void main(String[] args) {`
	
	`System.out.println("Welcome");`
	`displayInfo("Sara");`
	`displayInfo("Shahad");`
	`System.out.println("End");`
	`}`
	
	`public static void displayInfo(String name) {`
	
	`System.out.println("My Name is: " + name);`
	
	`}`	

2	`public static void main(String[] args) {`
	
	`int a;`
	
	`a = 5;`
	
	`a = square(a);`
	`System.out.println(a);`
	`}`
	
	`public static int square(int num) {`
	
	`return num * num;`
	
	`}`	

3

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

    int num = 4;

    if (isEven(num)) {
        System.out.println("it's even");
    } else {
        System.out.println("it's not even");
    }
}
public static boolean isEven(int num) {

    if (num % 2 == 0) {
        return true;
    }

    return false;

}
```

4

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

    int a = 4;

    int result = myMethod(a++);
    System.out.println("a= " + a);
    System.out.println("result=" + result);

}
public static int myMethod(int num) {

    System.out.println("num=" + num);

    return num++;
}
```

5

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

    for (int i = 0; i < 10; i++) {
        print(i);
    }
}
public static void print(int num) {

    System.out.println(num);
}
```

6

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

    int var = 0;

    for (int i = 1; i < 5; i++) {

        var += cal(i);
    }
    System.out.println("var=" + var);
}
public static int cal(int num) {
  return num * 2;

}
```

7

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

        System.out.println("start of main method..");
        method1();
        method2();
        System.out.println("end of main method");
    }
   private static void method1() {
        System.out.println("method1 executed.");
    }
    private static void method2() {
        System.out.println("method2 executed.");
    }
```

8

```
    public static void main(String[] args) {
        System.out.println("start");
        int a = 5;
        if(a< 5)
            method1();
        else
            method2();
        System.out.println("end");
    }
    public static void method1() {

        System.out.println("method1 executed.");
    }

    public static void method2() {

        System.out.println("method2 executed.");
    }
```

9

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

        System.out.println("start");
        int a = 5;
        int b = 3;
        int c;
        if (a >= 5) {
            c = method1(++b);
        } else {
            c = method2(++b);
        }
        System.out.println("a=" + a);
        System.out.println("b=" + b);
        System.out.println("c=" + c);
        System.out.println("end");

    }

    private static int method1(int a) {

        return a * a;
    }

    private static int method2(int a) {

        return a + a;

    }
```

10

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

    System.out.println("begin main");
    a();
    System.out.println("end main");

}

public static void a() {
    b();
    System.out.println("a");

}

public static void b() {
    System.out.println("b");
    c();
    d();
}

public static void c() {
    d();
    System.out.println("c");
}

public static void d() {

    System.out.println("d");
}
```

11

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

    System.out.println("begin main");

    int value = method(4 + 1);
    System.out.println("returned value is : " + value);
    System.out.println("end main");

}

public static int method(int a) {

    switch (a++) {

        case 6:
            return a = a + 10;

        case 5:
            return a = a * 10;

        default:
            return a = a - 10;
    }

}
```

12

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

    String s = "Sara";

    String result = "" + getChar(s, 2);
    result = result.concat(""+getChar(s, s.length()-1));
    System.out.println("result = " + result);

}
public static char getChar(String str, int index) {

    return str.charAt(index);

}
```

13

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

    String s = method("COMPUTER");
    System.out.println(s);

    s = method("ORACLE");
    System.out.println(s);

    s = method("NAIF");
    System.out.println(s);

}
public static String method(String str) {

    String var = "";

    for (int i = str.length() - 1; i > -1; i--) {

        var = var.concat("" + str.charAt(i));
    }
    return var;
}
```

14

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

    String s = method("MobilePhone");
    System.out.println(s);

    s = method("University");
    System.out.println(s);
}
public static String method(String s) {
    switch (s.substring(8)) {
        case "Phone":
            return s.substring(5, s.length() - 4);

        case "hone":
            return s.substring(2, s.length() - 1);

        case "one":
            return s.substring(4, s.length() - 3);
    }
    return "Oops!";
}
```

[Coding Questions Part#1]

Q1.Write a method called printUp that takes an integer as a parameter and then the method should print number from 1 up to the passed number.
[Hint: Test your method from the main method].

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

    printUp(10);   //method call
}

public static  void printUp(int num){


    for (int x=1 ; x<=num ; x++){
        System.out.println(x);
    }
}
```

Q2.Write a method called multiply that takes two integers as parameters and then your method should return the product of the two numbers using addition operation only.
[Hint: Test your method from the main method].

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

    int result = multiply(3 , 5);  // call the method and store the returned value in variable result.

    System.out.println("Product = "+result );
}

public static  int multiply(int a , int b){

    int product = 0;

    for (int x=1 ; x<=b ; x++){
        product +=a;
    }

    return product;
}
```

Q3.Write a method called power that takes two integers as parameters and then your method should calculate the power of first parameter as a base to the second parameter which is the exponent.(i.e a^b , a is the base , b is the exponent)
[Hint: Test your method from the main method].

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

    int result = power(4 , 2); // method call

    System.out.println("Product = "+result );
}

public static  int power(int base , int exp){

    int power = 1;

    for (int x=1 ; x<=exp ; x++){
        power *= base;
    }

    return power;
}
```

Q4.Write a method called printEvens that prints even numbers from 1 up to 20 using while loop.
[Hint: Test your method from the main method].

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

    printEvens();
}

public static  void printEvens(){
    int x=1 ;

     while ( x<=20 ){

         if(x % 2 ==  0)
        System.out.println(x);

         x++;
    }
}
```

Q5. Modify the above program to print ODD numbers instead of evens.
Q6. Modify the above program to print numbers greater than 5 only.
Q7. Modify the above program to print even numbers from 1 up to the number the user will pass in the method parameter list.

Q8.Write a method called findSum that takes one parameter called num where the method should calculate the sum of numbers from 1 up to num
[Hint: Test your method from the main method].

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

    System.out.println("Sum is : "+findSum(5));
}

public static  int findSum(int num){

    int sum=0;

    for (int x=1 ; x<=num ; x++){
        sum+=x;
    }
    return sum;
}
```

Q8.Write a method called checkEven which takes an integer as a parameter and then return true if the number is divisible by 3 , false otherwise.
[Hint: Test your method from the main method].

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

   boolean result = checkDivisibleBy3(4);

    System.out.println((result==true)?"it's divisible by 3":"it's not divisible by 3");
}

public static  boolean checkDivisibleBy3(int num){

    if(num%3==0)
        return true;

    else
        return false;
}
```

Q9.Write a method to check if the passed string begins with a vowel letter or not .
Q10. Write a Java method to display the middle character of a string
Q11. Write a Java method to compute the sum of the digits in an integer
Q12. Write a Java method to return the revers for the passed integer .

Good Luck
Written with ♥ By :
ENG.NAIF ALSHEHRI