1. What is the data-type usually used in arithmetic operation?

The Java programming language supports various arithmetic operators for all floating-point and integer numbers. These operators are + (addition), - (subtraction), * (multiplication), / (division), and % (modulo). The following table summarizes the binary arithmetic operations in the Java programming language.

Operator	Use	Description
+	a1 + b2	Adds a1 and b2; also used to concatenate strings
-	a1 - b2	Subtracts b2 from a1
*	a1 * b2	Multiplies a1 by b2
/	a1 / b2	Divides a1 by b2
%	a1 % b2	Computes the remainder of dividing a1 by b2

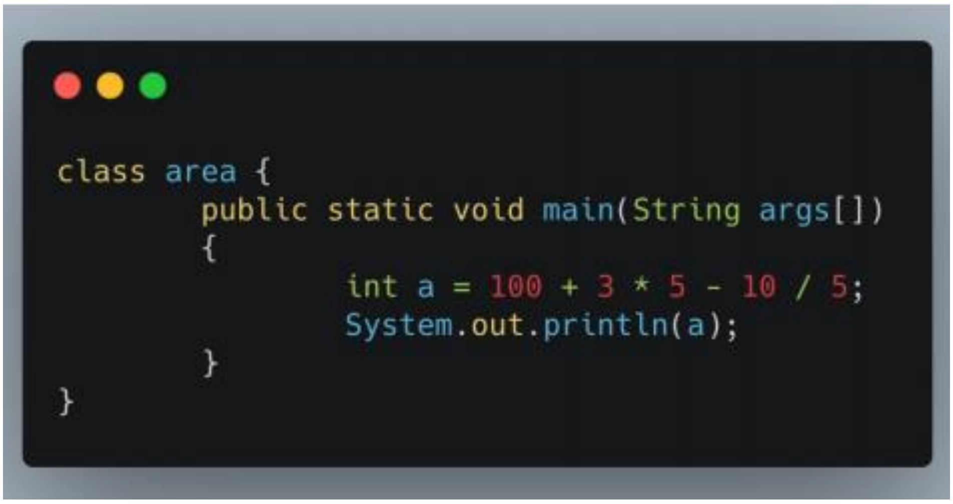
2. What is the data-type usually used in word and character?

Language	Reserved Word	Example
C++	char	'A'
C++	string	"Hello world!"
C#	char	'A'
C#	String	"Hello world!"
Java	char	'A'
Java	String	"Hello world!"
JavaScript	String	'Hello world!', "Hello world!"
Python	str()	'Hello world!', "Hello world!"
Swift	Character	"A"
Swift	String	"Hello world!"

a. **character**
A data type representing single text characters like the alphabet, numeral digits, punctuation, etc.

b. **string**
A series or array of characters as a single piece of data.

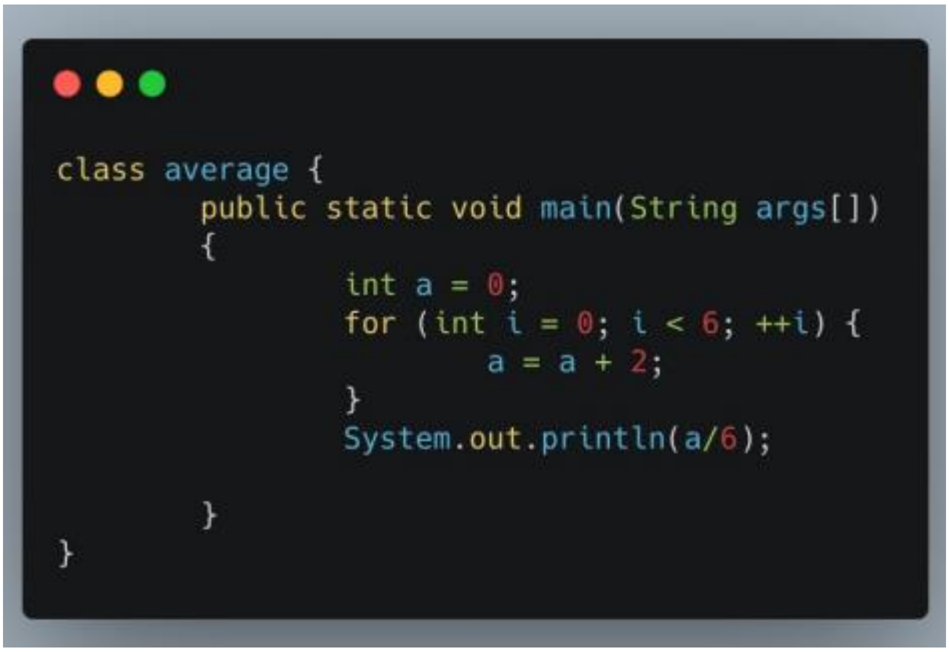
3. What is the output of code below:



```
class area {
        public static void main(String args[])
        {
                int a = 100 + 3 * 5 - 10 / 5;
                System.out.println(a);
        }
}
```

Output: 113

4. What is the output of code below:

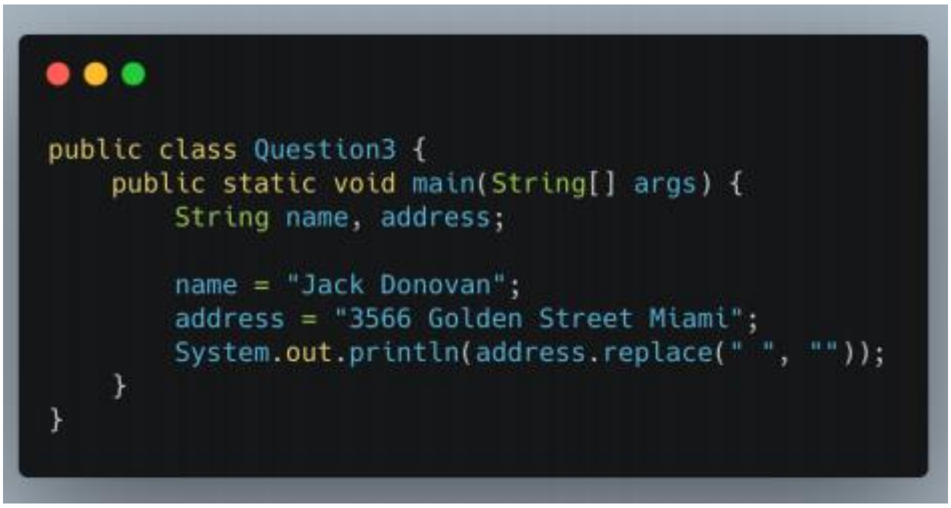


```
class average {
        public static void main(String args[])
        {
                int a = 0;
                for (int i = 0; i < 6; ++i) {
                        a = a + 2;
                }
                System.out.println(a/6);

        }
}
```

Output: 2

5. **What is the output of code below:**



```java
public class Question3 {
    public static void main(String[] args) {
        String name, address;

        name = "Jack Donovan";
        address = "3566 Golden Street Miami";
        System.out.println(address.replace(" ", ""));
    }
}
```

Output: 3566GoldenStreetMiami

Flaky Test

1. **What is a flaky test?**

A flaky test is a test that's unreliable in behaviour, meaning that it yields different results inconsistently.

One moment it will pass (and probably did when it was merged to the codebase), and the next it will suddenly fail, perhaps only to then pass again when re-run.

They are sometimes referred to as "random failures", but in reality, it's often less about actual randomness than very reproducible edge cases that happen in a *seemingly random fashion*.

2. **An element in a website is using ajax/javascript to show the data (processed as asynchronous). What do you do to test that element?**

   The **biggest challenge in handling Ajax call is knowing the loading time for the web page.** Since the loading of the web page will last only for a fraction of seconds, it is difficult for the tester to test such application through automation tool. For that, Selenium Webdriver has to use the wait method on this Ajax Call.

So by executing this wait command, selenium will suspend the execution of current Test Case and wait for the expected or new value. When the new value or field appears, the suspended test cases will get executed by Selenium Webdriver.

Following are the wait methods that Selenium Webdriver can use:

1. **Thread.Sleep()**

- Thread.Sleep () is not a wise choice as it suspends the current thread for the specified amount of time.
- In AJAX, you can never be sure about the exact wait time. So, your test will fail if the element won't show up within the wait time. Moreover, it increases the overhead because calling Thread.sleep(t) makes the current thread to be moved from the running queue to the waiting queue.
- After the time 't' reached, the current thread will move from the waiting queue to the ready queue, and then it takes some time to be picked by the CPU and be running.

2. **Implicit Wait()**

This method tells webdriver to wait if the element is not available immediately, but this wait will be in place for the entire time the browser is open. So any search for the elements on the page could take the time the implicit wait is set for.

3. **Explicit Wait()**

Explicit wait is used to freeze the test execution till the time a particular condition is met or maximum time lapses.

4. **WebdriverWait**

- It can be used for any conditions. This can be achieved with WebDriverWait in combination with ExpectedCondition

- The best way to wait for an element dynamically is checking for the condition every second and continuing to the next command in the script as soon as the condition is met.

But the problem with all these waits is, you have to mention the time out unit. What if the element is still not present within the time? So there is one more wait called Fluent wait.

5. **Fluent Wait**

This is an implementation of the Wait interface having its timeout and polling interval. Each FluentWait instance determines the maximum amount of time to wait for a condition, as well as the frequency with which to check the condition.

3. A website page can only be accessed with CAPTCHA. How do you test that page?

But what happens when you need to do performance testing on such a site? One common question on our support line is how to configure a load testing tool to read the displayed text and type it in. The whole point of adding captcha security is to prevent an automated tool from accessing the website, so if it was easily bypassed by a load testing tool, then spammers could also use that same technique to access your website!

There are three options that I can see:

1. Completely disable captcha for the purposes of the test
2. Configure the captcha software to accept anything that is typed as being valid
3. Configure the captcha to display the actual value in the image somewhere in the web page where it can be parsed by Load Tester. (Obviously this must be removed before shipping.)