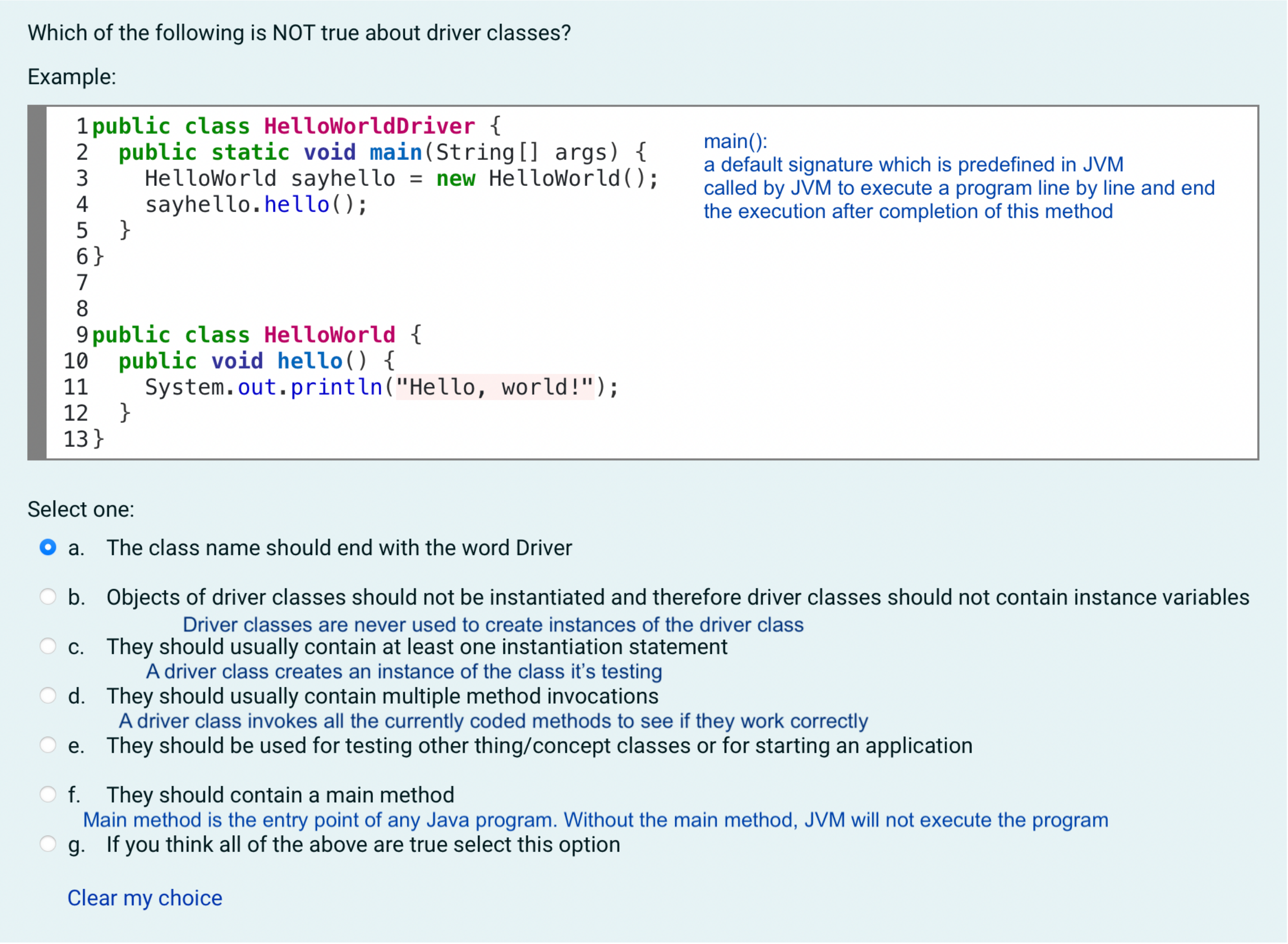
Driver class



Which of the following is NOT true about driver classes?

Example:

```
public class HelloWorldDriver {
  public static void main(String[] args) {
    HelloWorld sayhello = new HelloWorld();
    sayhello.hello();
  }
}


public class HelloWorld {
  public void hello() {
    System.out.println("Hello, world!");
  }
}
```

main():
a default signature which is predefined in JVM
called by JVM to execute a program line by line and end the execution after completion of this method

Select one:

● a. The class name should end with the word Driver

○ b. Objects of driver classes should not be instantiated and therefore driver classes should not contain instance variables

Driver classes are never used to create instances of the driver class

○ c. They should usually contain at least one instantiation statement

A driver class creates an instance of the class it's testing

○ d. They should usually contain multiple method invocations

A driver class invokes all the currently coded methods to see if they work correctly

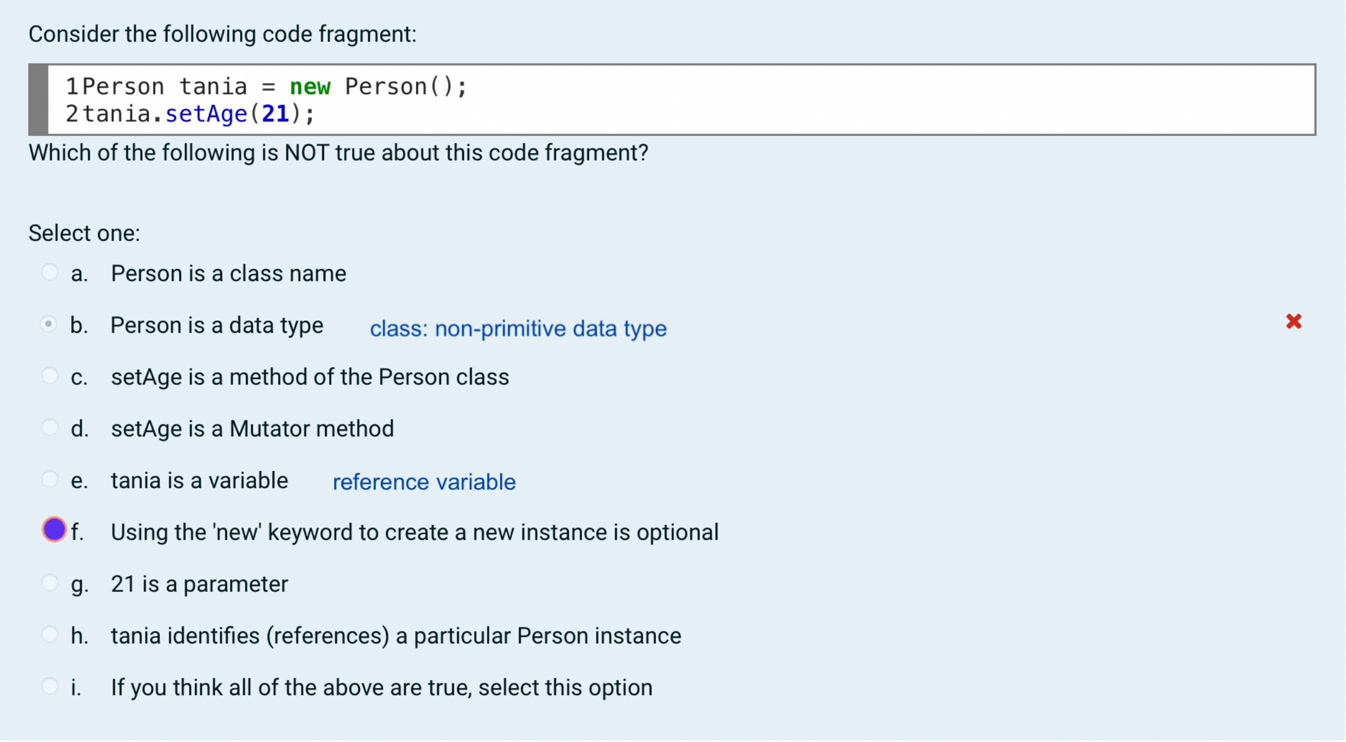
○ e. They should be used for testing other thing/concept classes or for starting an application

○ f. They should contain a main method

Main method is the entry point of any Java program. Without the main method, JVM will not execute the program

○ g. If you think all of the above are true select this option

Clear my choice



Consider the following code fragment:

```
Person tania = new Person();
tania.setAge(21);
```

Which of the following is NOT true about this code fragment?

Select one:

○ a. Person is a class name

● b. Person is a data type   class: non-primitive data type ✖

○ c. setAge is a method of the Person class

○ d. setAge is a Mutator method

○ e. tania is a variable   reference variable

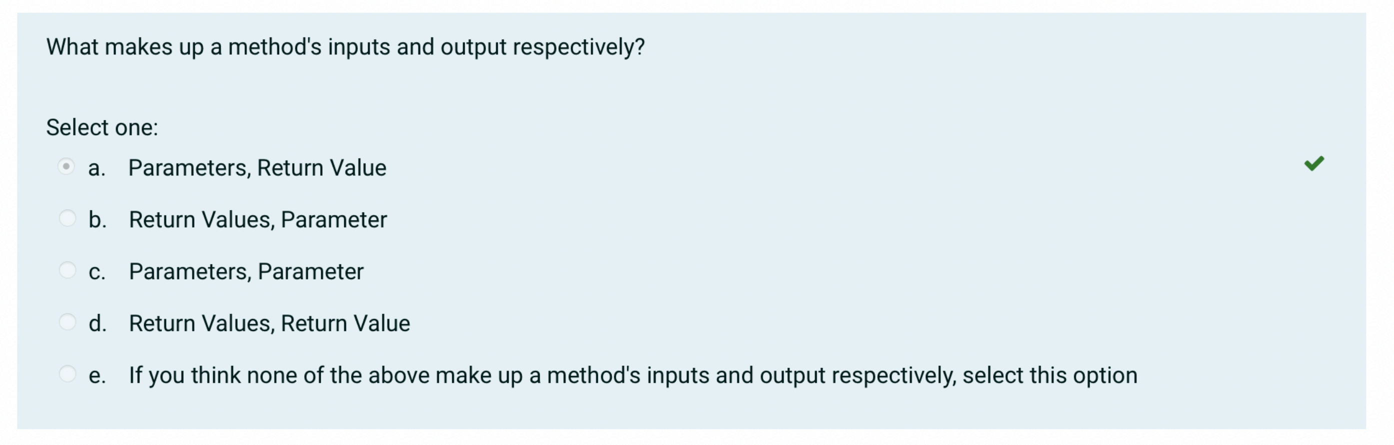
● f. Using the 'new' keyword to create a new instance is optional

○ g. 21 is a parameter

○ h. tania identifies (references) a particular Person instance

○ i. If you think all of the above are true, select this option

Method



What makes up a method's inputs and output respectively?

Select one:

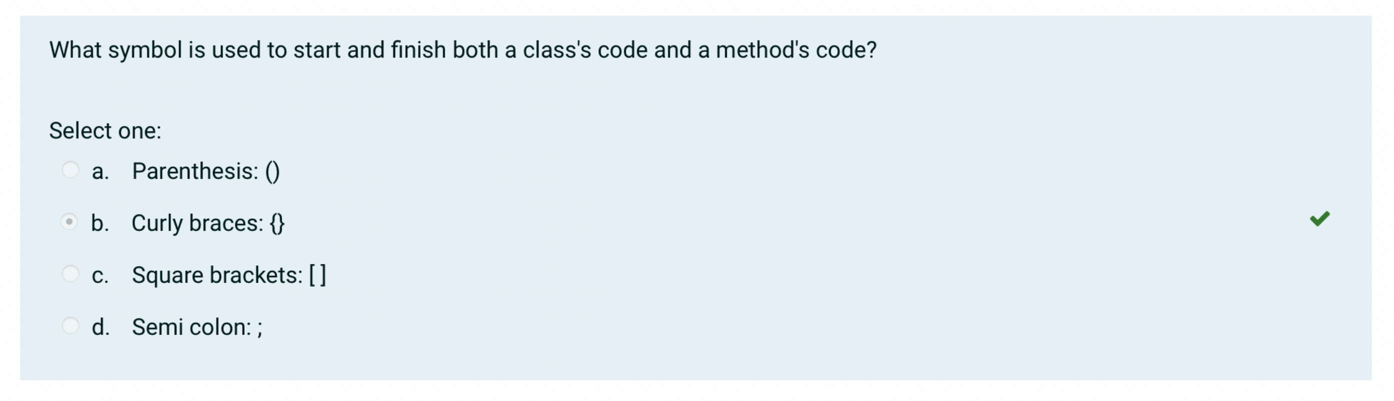
● a. Parameters, Return Value ✓

○ b. Return Values, Parameter

○ c. Parameters, Parameter

○ d. Return Values, Return Value

○ e. If you think none of the above make up a method's inputs and output respectively, select this option



What symbol is used to start and finish both a class's code and a method's code?

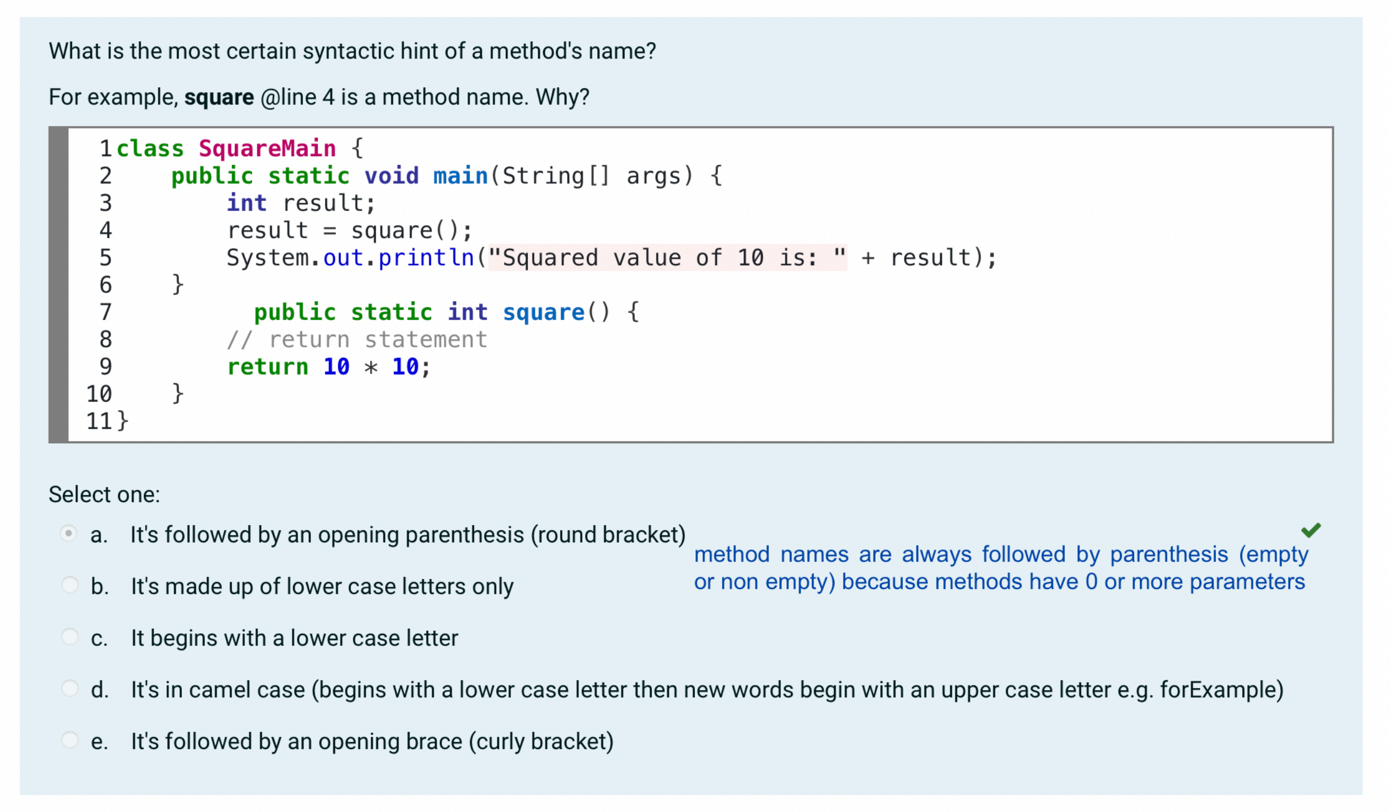
Select one:

○ a. Parenthesis: ()

● b. Curly braces: {} ✓

○ c. Square brackets: [ ]

○ d. Semi colon: ;



What is the most certain syntactic hint of a method's name?

For example, **square** @line 4 is a method name. Why?

```
 1class SquareMain {
 2    public static void main(String[] args) {
 3        int result;
 4        result = square();
 5        System.out.println("Squared value of 10 is: " + result);
 6    }
 7          public static int square() {
 8        // return statement
 9        return 10 * 10;
10    }
11}
```

Select one:

● a. It's followed by an opening parenthesis (round bracket) ✓

method names are always followed by parenthesis (empty or non empty) because methods have 0 or more parameters

○ b. It's made up of lower case letters only

○ c. It begins with a lower case letter

○ d. It's in camel case (begins with a lower case letter then new words begin with an upper case letter e.g. forExample)

○ e. It's followed by an opening brace (curly bracket)

Which of the following is NOT true?

Select one:

○ a. Accessor methods must include a return statement with a return expression

only task is to retrieve the current value of an instance variable

Example:

```java
private String myField; //"private" means access to this is restricted

public String getMyField()
{
    //include validation, logic, logging or whatever you like here
   return myField;
}
```

○ b. Mutator methods must have an input parameter

Example:

```java
private String myField; //"private" means access to this is restricted


public void setMyField(String value)
{
    //include more logic
    myField = value;
}
```

○ c. Mutator method code is responsible for maintaining the integrity of an object's data

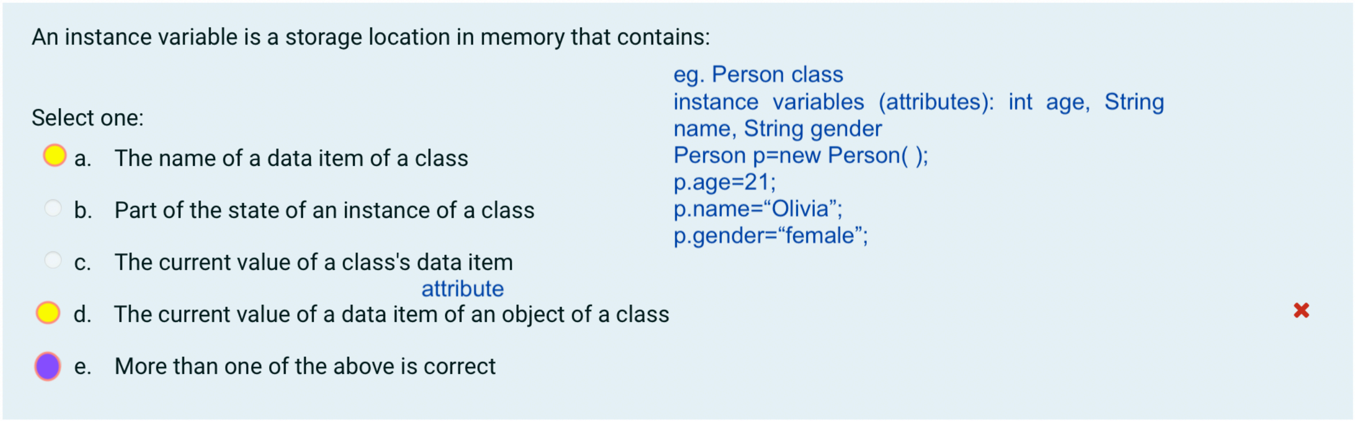
only task is to reset the value of an instance

○ d. Accessor and Mutator methods are always public because they are intended to be used by code outside their class

● e. Mutator methods must include a return statement with a return expression ✓

○ f. If you think all of the above are true, select this option

Instance variable



An instance variable is a storage location in memory that contains:

eg. Person class
instance variables (attributes): int age, String name, String gender
Person p=new Person( );
p.age=21;
p.name="Olivia";
p.gender="female";

Select one:

○ a. The name of a data item of a class

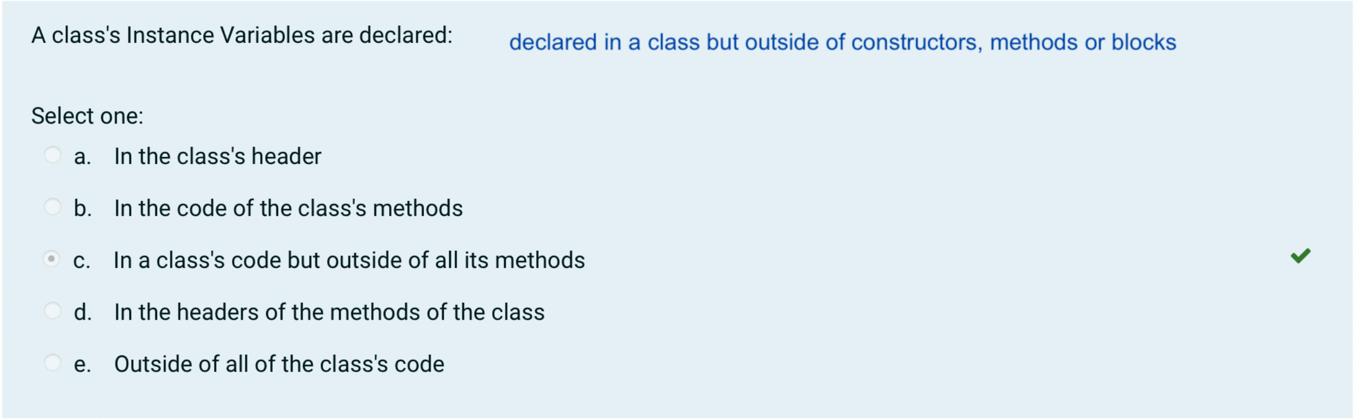
○ b. Part of the state of an instance of a class

○ c. The current value of a class's data item

attribute

○ d. The current value of a data item of an object of a class ✗

● e. More than one of the above is correct



A class's Instance Variables are declared: declared in a class but outside of constructors, methods or blocks

Select one:

○ a. In the class's header

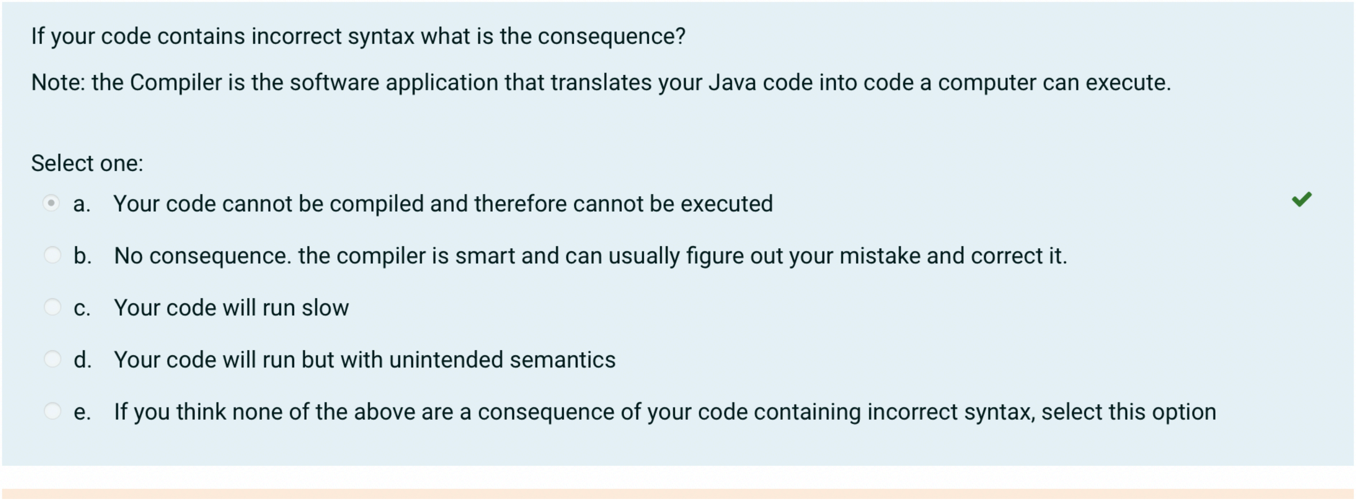
○ b. In the code of the class's methods

● c. In a class's code but outside of all its methods ✓

○ d. In the headers of the methods of the class

○ e. Outside of all of the class's code

Syntax error



If your code contains incorrect syntax what is the consequence?

Note: the Compiler is the software application that translates your Java code into code a computer can execute.

Select one:

● a. Your code cannot be compiled and therefore cannot be executed ✓

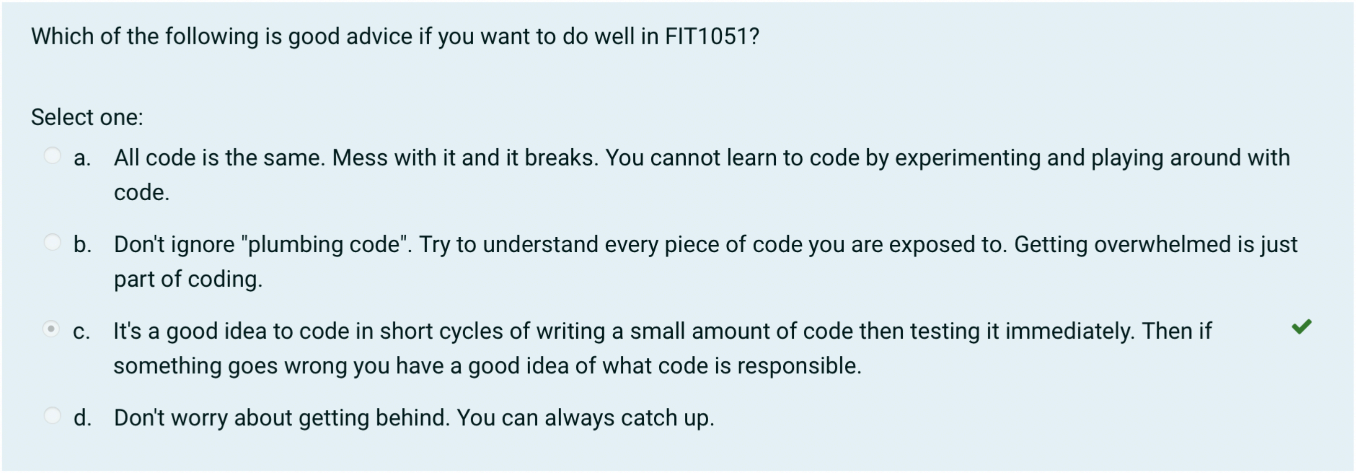
○ b. No consequence. the compiler is smart and can usually figure out your mistake and correct it.

○ c. Your code will run slow

○ d. Your code will run but with unintended semantics

○ e. If you think none of the above are a consequence of your code containing incorrect syntax, select this option

Course study



Which of the following is good advice if you want to do well in FIT1051?

Select one:

○ a. All code is the same. Mess with it and it breaks. You cannot learn to code by experimenting and playing around with code.

○ b. Don't ignore "plumbing code". Try to understand every piece of code you are exposed to. Getting overwhelmed is just part of coding.

● c. It's a good idea to code in short cycles of writing a small amount of code then testing it immediately. Then if something goes wrong you have a good idea of what code is responsible. ✓

○ d. Don't worry about getting behind. You can always catch up.