Object Orientated Programming VIVA

NAME: VIKHYAT BANSAL
ROLLNO: CB.EN.U4AIE21076

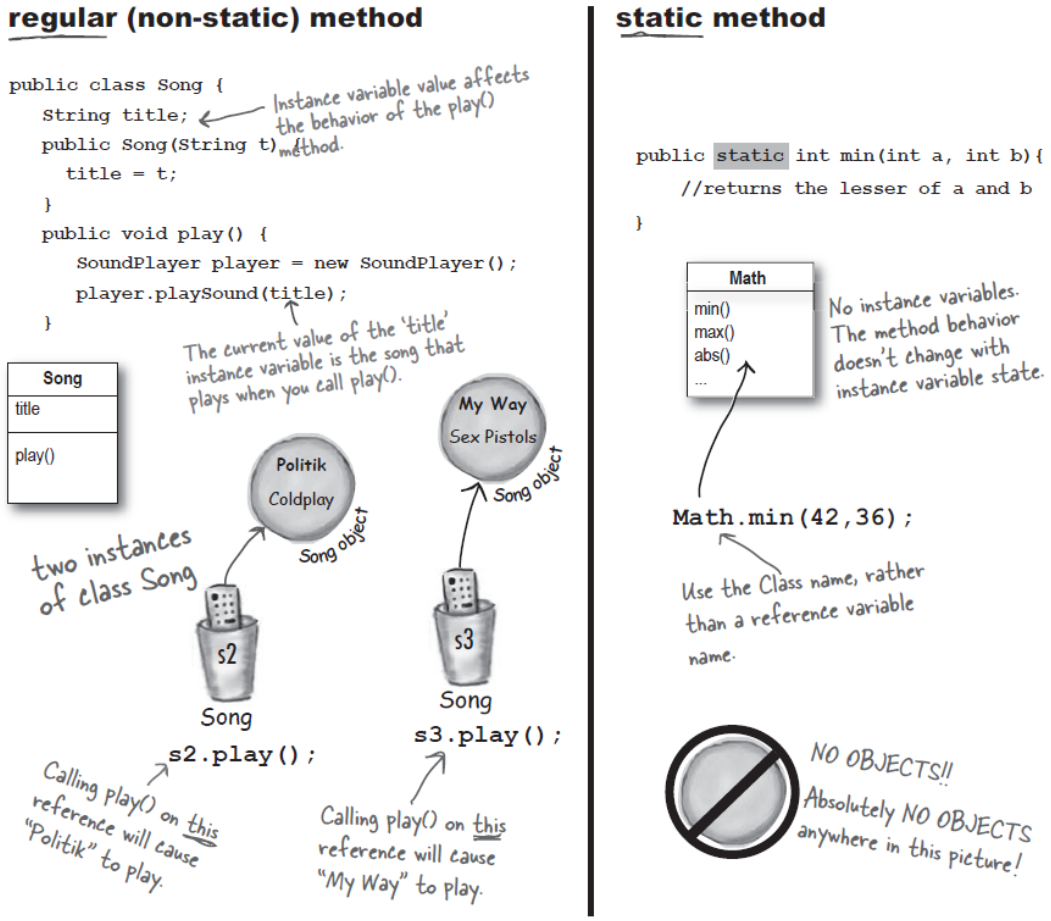
Q1. What is keyword static in JAVA?

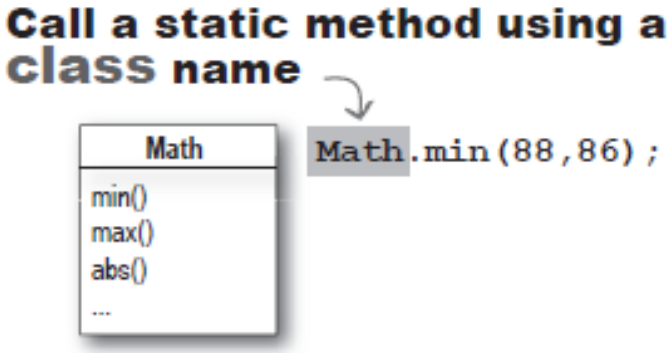
Answer I have given :-

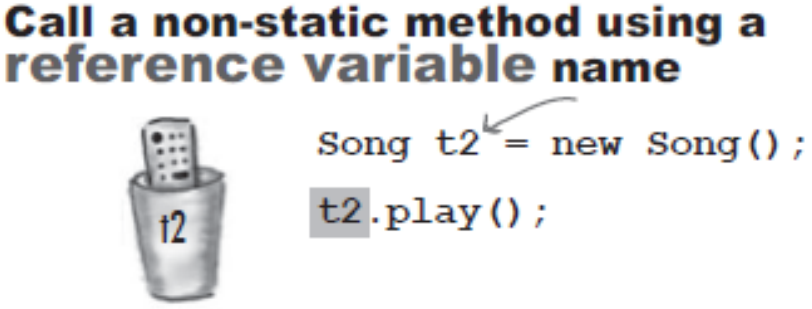
The static keyword in Java is used for memory management mainly. We can apply static keyword with variables, methods, blocks and nested classes. The static keyword belongs to the class than an instance(object) of the class.

ANSWER WHICH I HAVE COME ACROSS:-

The keyword static lets a method run without any instance of the class. A static method means “behaviour not dependent on an instance variable, so no instance/object is required. Just the class.”







From above picture it can be seen that an object is being created and then called in non-static method .
In case of static method no objects or instance are being created and we use the name of class for calling function.

Advantages of static variable

It makes your program memory efficient (i.e., it saves memory).

EXAMPLE

Suppose there are 500 students in my college, now all instance data members will get memory each time when the object is created. All students have its unique roll no. and name, so instance data member is good in such case. Here, "college" refers to the common property of all objects. If we make it static, this field will get the memory only once.

Common Doubt

Q: I could swear I've seen code that calls a static method using a reference variable instead of the class name.

A: You *can* do that, but as your mother always told you, "Just because it's legal doesn't mean it's good." Although it *works* to call a static method using any instance of the class, it makes for misleading (less-readable) code. You *can* say,

```
Duck d = new Duck();
String[] s = {};
d.main(s);
```

This code is legal, but the compiler just resolves it back to the real class anyway ("OK, *d* is of type Duck, and main() is static, so I'll call the static main() in class Duck"). In other words, using *d* to invoke main() doesn't imply that main() will have any special knowledge of the object that *d* is referencing. It's just an alternate *way* to invoke a static method, but the method is still static!