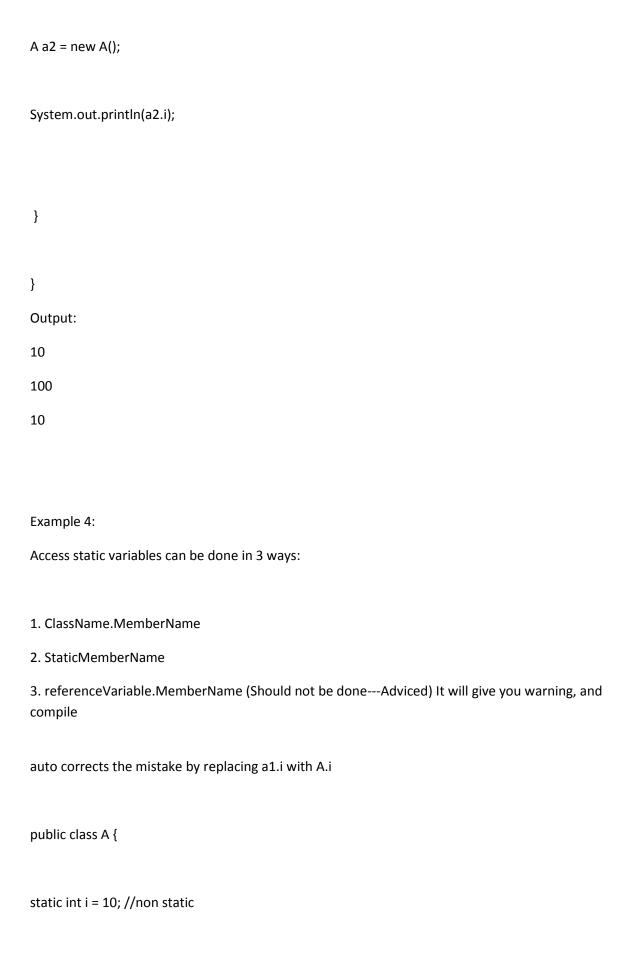
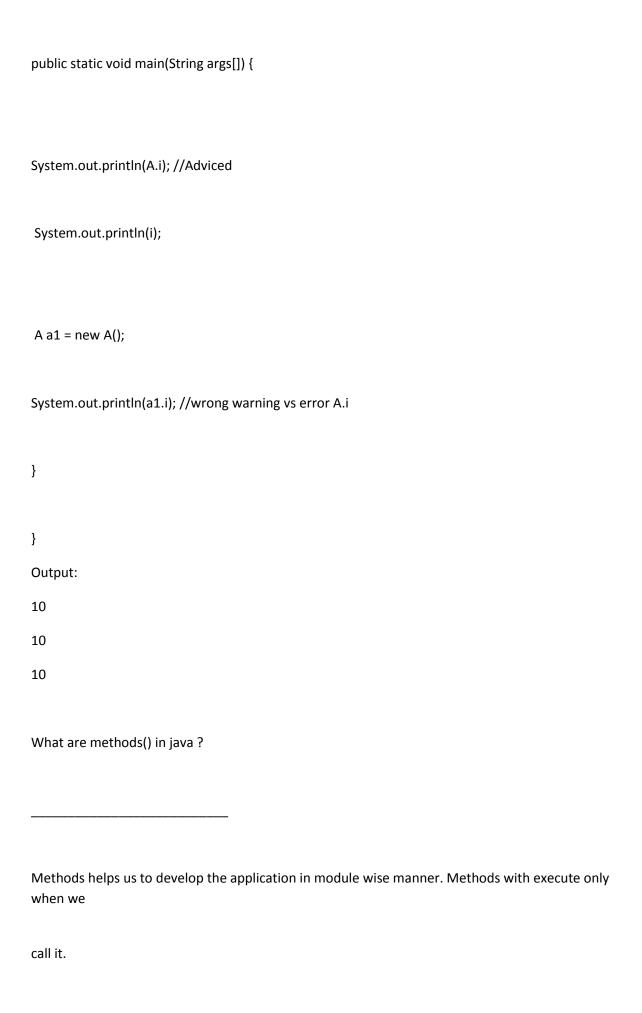
non-static versus static members:
1. These members belongs to Object and once object is created it will be loaded into the object
2. Syntax: referenceVariable.objectMemberName
3. Every time we create an object a copy of non-static member will get loaded into it
4. Every object has its own copy of non-static member. Hence making changes to non static member of
one object will not create any impact on non static member of other object
Static Memeber:
1. These members belonds to class common memory and is loaded in to the common memory only once.
2. Syntax to access that is ClassName.Membername
Example 1:
public class A {
static int i = 100;
int j = 500; //non static
public static void main(String args[]) {
A a1 = new A();
System.out.println(a1.j);

```
System.out.println(A.i);
}
}
Output:
500
100
Example 2:
public class A {
static int i = 100;
int j = 500; //non static
 public static void main(String args[]) {
 A a1 = new A();
  A a2 = new A();
System.out.println(a1.j);
System.out.println(a2.j);
```

```
System.out.println(A.i);
}
}
Ouput:
500
500
100
Example 3:
public class A {
int i = 10; //non static
public static void main(String args[]) {
A a1 = new A();
System.out.println(a1.i);
a1.i = 100;
System.out.println(a1.i);
```





```
If a method is non static then we should create object and only then call that. But if the method is
static the we call that with class name, here object creation is not required.
Example 1:
public class A {
public static void main(String args[]) {
A a1 = new A();
}
public void test(){
System.out.println("From test");
}
}
Output:
From test
Example 2:
public class A {
```

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

A.test();
}

public static void test(){

System.out.println("From test");
}

Output:
```

From test