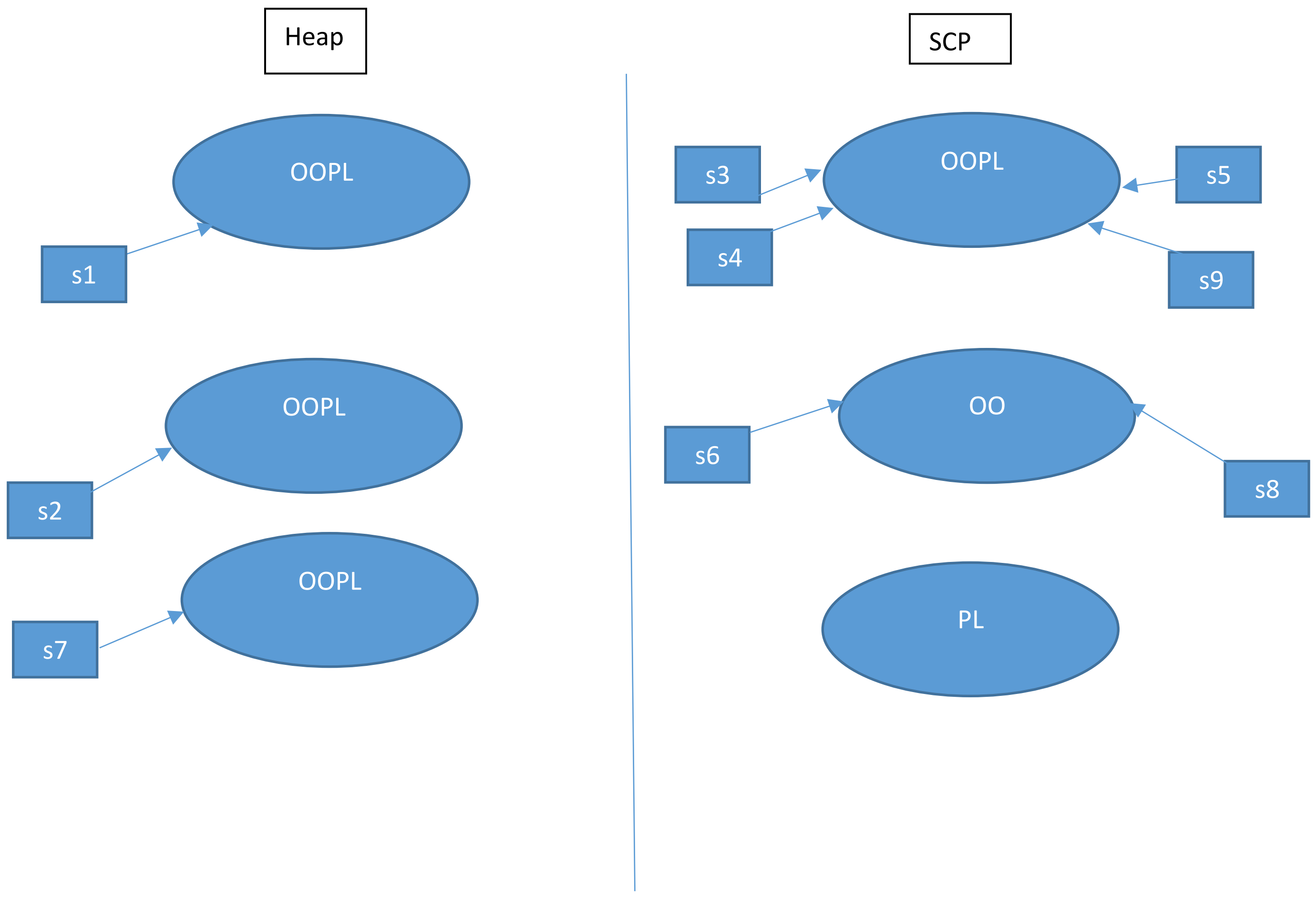
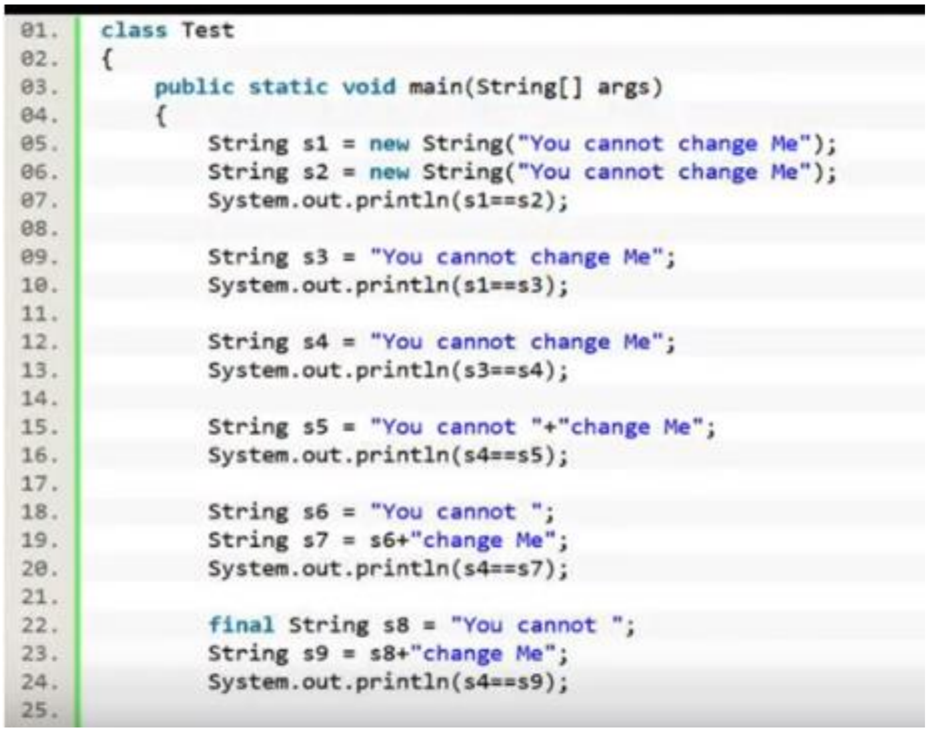
Program Code:

1. String s1= new String(“Object Oriented Programming Language”);//OOPL
2. String s2= new String(“Object Oriented Programming Language”);/OOPL
3. String s3= “Object Oriented Programming Language”;
4. String s4= “Object Oriented Programming Language”;
5. String s5= “Object Oriented” + “Programming Language”;//→”OOPL “
6. String s6= “Object Oriented”;
7. String s7= s6 + “Programming Language”;
8. final String s8= “Object Oriented”;
9. String s9= s8+ “Programming Language”;
10. SOP(s1==s2);//false
11. SOP(s2==s3);//
12. SOP(s3==s4);//
13. SOP(s3==s5);//
14. SOP(s4==s5);//
15. SOP(s7==s9);//
16. SOP(s4==s9);//





```
01. class Test
02. {
03.     public static void main(String[] args)
04.     {
05.         String s1 = new String("You cannot change Me");
06.         String s2 = new String("You cannot change Me");
07.         System.out.println(s1==s2);
08.
09.         String s3 = "You cannot change Me";
10.         System.out.println(s1==s3);
11.
12.         String s4 = "You cannot change Me";
13.         System.out.println(s3==s4);
14.
15.         String s5 = "You cannot "+"change Me";
16.         System.out.println(s4==s5);
17.
18.         String s6 = "You cannot ";
19.         String s7 = s6+"change Me";
20.         System.out.println(s4==s7);
21.
22.         final String s8 = "You cannot ";
23.         String s9 = s8+"change Me";
24.         System.out.println(s4==s9);
25.
```

5: 1 UCCM in Heap+1 UCCM in SCP----2 objects
6: 1 UCCM in Heap +0 bcz already in SCP
7: both are pointing to different object----false
9: thru this line one UCCM is required to create in SCP area but in SCP area UCCM is already there so now s3 →UCCM in scp.
10: s1 and s3 both are pointing to different objects----false.
12: thru this line one UCCM is required to create in SCP area but in SCP area UCCM is already there so now s4 →UCCM in heap.
13: both are pointing to same object present in SCP ---true

15: two object will be created in SCP   YC and CM ----- Wrong
and one object will be created in Heap area due to runtime operation------Wrong

Reason: It is a compile time operation not a run time operation.
SOP(10+20);
JVM will print 30 directly because both are constant and this operation will be performed only at compile time not required to wait until run time.
so in line no 15 both String objects are String literals or String constant so this concatenation will be performed at compile time not run time, so at run time UCCM content come like a literal constant ,so output will be UCCM it is similar to line no 9 and 12.
so now onward s5 is also pointing to UCCM in SCP s5→UCCM

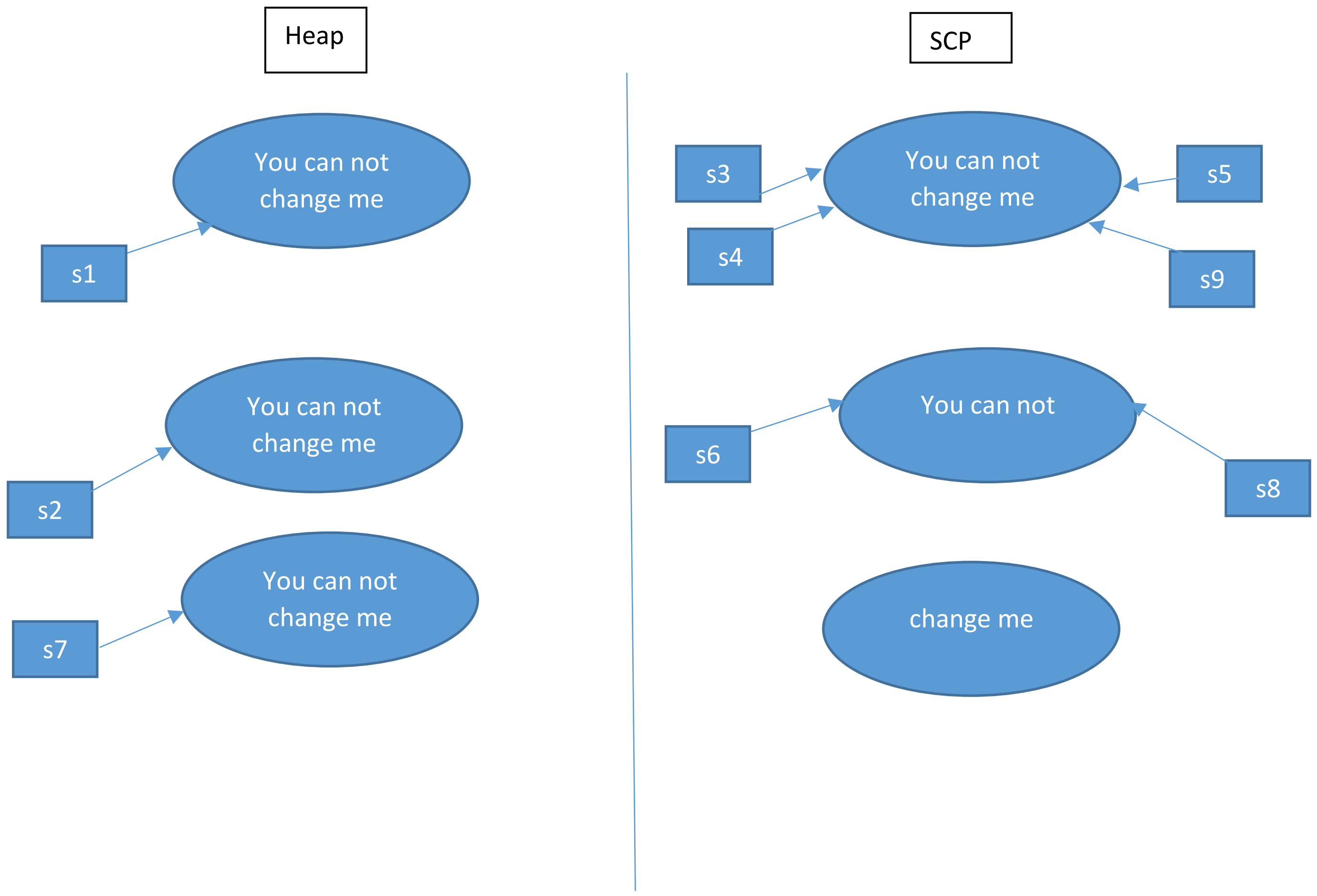
16; both are pointing to same object---true

18: UC created in SCP area   s6→UC
19: normal var(s6) + String constant -- behave like a run time operation
so CM in SCP and UCCM in heap s7→ UCCM in heap.

20: both are pointing to different object –false

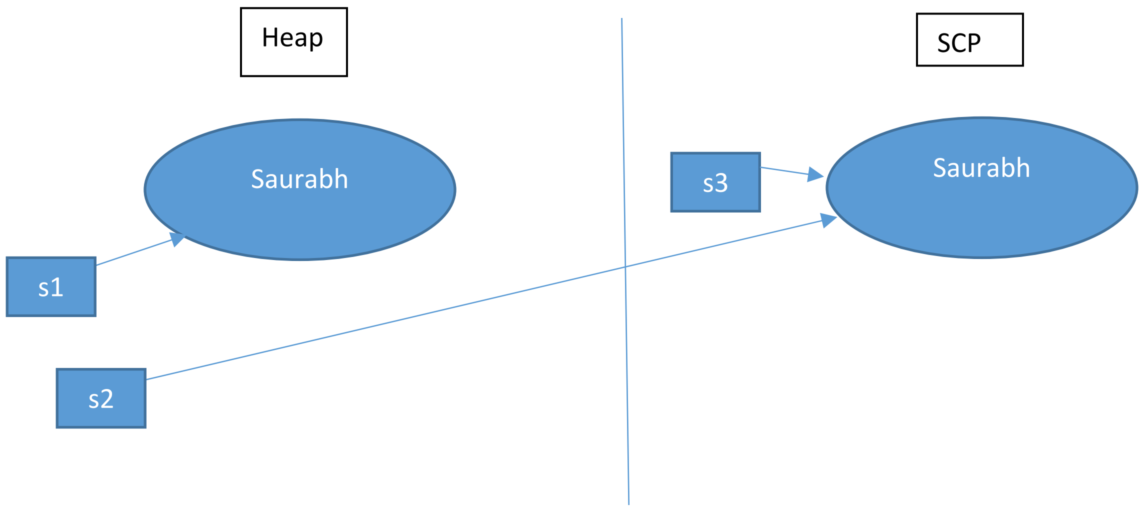
22: UC is already present in SCP so s8→UC in SCP
23: Every final variable will be replaced at compile time, and final variable is constant variable
so   s8(constant)+ CM(constant) operarion performed at compile time like no 15.
so now onwards s9 is pointing to UCCM in SCP area.s9→UCCM

24: bot are pointing same object—true.



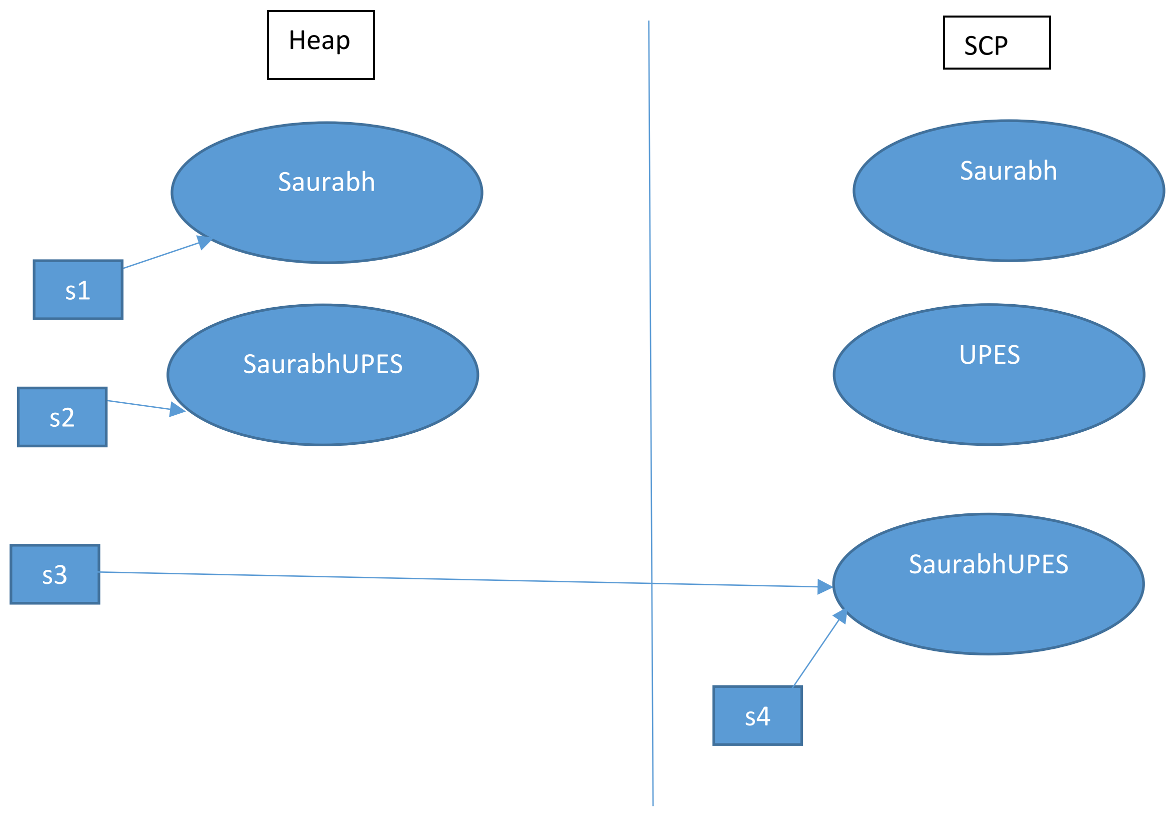
public String intern()
It can be used to return string from memory, if it is created by new keyword. It creates exact copy of heap string object in string constant pool.
By Using the heap object reference if you want to get corresponding SCP object then we should go for intern() method.

```
String s1 = new String(" Saurabh");
String s2= s1.intern();//
System.out.println(s1==s2);//false
String s3= "Saurabh";
System.out.println(s2==s3);//True
```



If the corresponding object not available in SCP then intern() creates that objects and returns it.

```
String s1 = new String(" Saurabh");
String s2= s1.concat("UPES");
String s3= s2.intern();//create and returns string
String s4= "SaurabhUPES";
System.out.println(s3==s4);//True
```



```
public class InternExample{
public static void main(String args[]){
String s1=new String("UPES");
String s2="UPES";
String s3=s1.intern();//returns string from pool, now it will be same as s2
System.out.println(s1==s2);//false because reference variables are pointing to different instance
System.out.println(s2==s3);//true because reference variables are pointing to same instance
}}
```

```
false
true
```

```
public class InternExample2 {
    public static void main(String[] args) {
        String s1 = "Saurabh";
        String s2 = s1.intern();
        String s3 = new String("Saurabh");
        String s4 = s3.intern();
        System.out.println(s1==s2); // True
        System.out.println(s1==s3); // False
        System.out.println(s1==s4); // True
        System.out.println(s2==s3); // False
        System.out.println(s2==s4); // True
        System.out.println(s3==s4); // False
    }
}
```

Test it Now

```
true
false
true
false
true
false
```

Some Methods of String class:

1. **public char charAt(int index)**

   Example:
   String s= “Saurabh”;
   SOP(s.charAt(2));//u
   SOP(s.charAt(6));//h
   SOP(s.charAt(10));//RE :StringIndexOutOfBoundException

2. **public String concat(String s);**

   String s= “Saurabh”;
   s=s.concat(“UPES”);
   OR
   //s=s + “UPES”;// overloaded + is also used for String Concatation
   //s+= “UPES”;//+= is also……
   SOP(s);//SaurabhUPES

3. **public boolean equals(Object ob):** used for content comparison where the case is also important
4. **public boolean equalsIgnoreCase(String s):** used for content comparison where the case is also not important.

String s = “UPES”;
SOP(s.equals(“upes”));//false
SOP(s.equalsIgnoreCase(upes));//true

5. **public String substring(int begin)**;// returns the substring from begin index to end of the string.

6. **public String substring(int begin, int end)**; /returns the substring from begin index to end-1 index.

Example:
String s= “Language”;
SOP(s.substring(4));// uage
SOP(s.substring(2,5)); // ngu

7. public int length();

String s= “Saurabh”;
SOP(s.length()); // 7
SOP(s.length); // CE

Note: length variable is applicable for arrays where as length() method is applicable for String objects.

8. String replace(char old, char new);
Ex:
String s= “aabbb”;
SOP(s.replace(‘a’ , ‘b’); // bbbbb

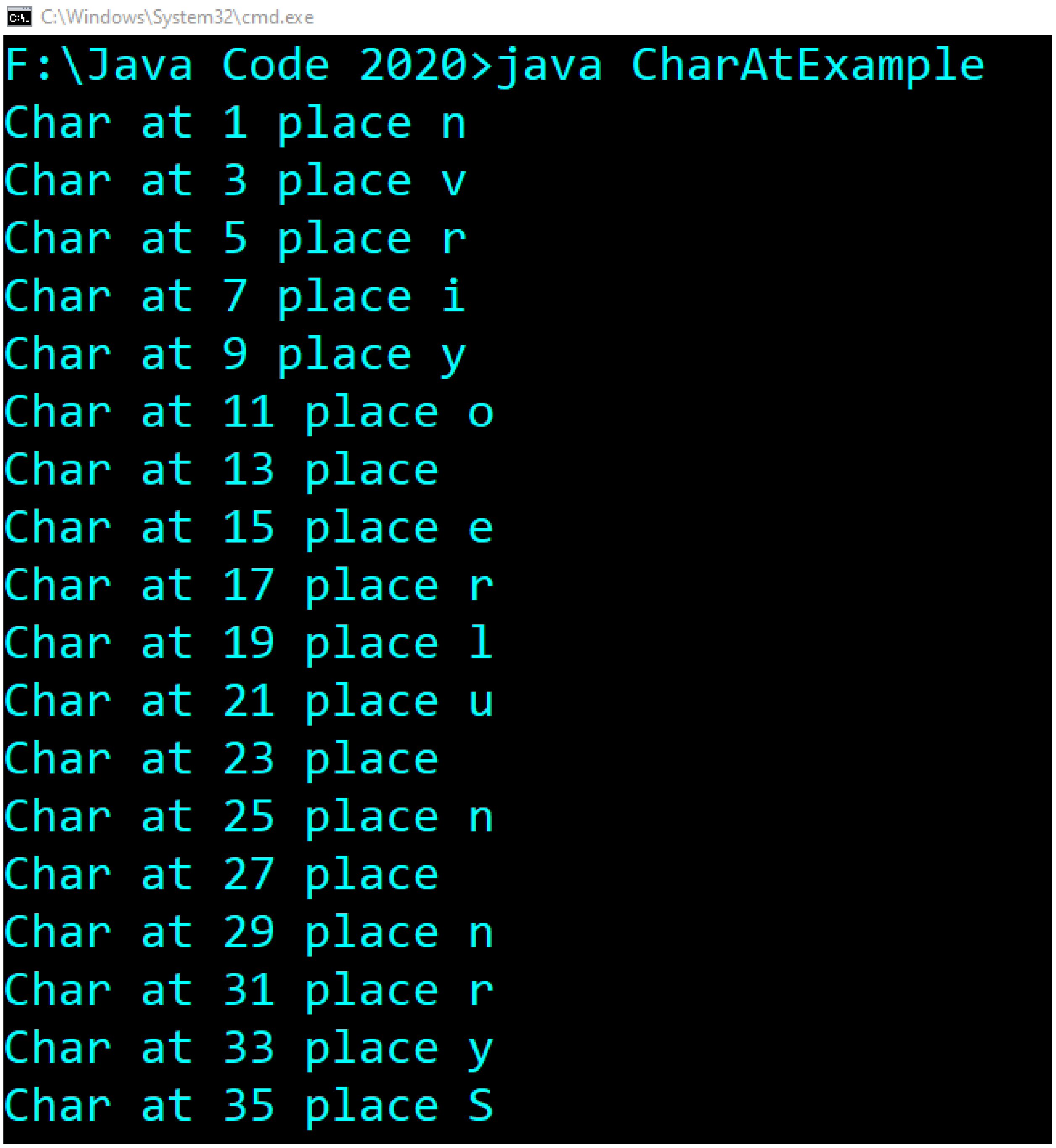
9. public String toLowerCase();
10. Public String toUpperCase();

11. public String trim();
to remove the blank spaces at beginning and end of the string, both not blanks paces present at middle of the string.

12. public int indexOf(char ch):
It returns index of first occurrence of the specified character.

Let's see an example where we are accessing all the elements present at odd index.

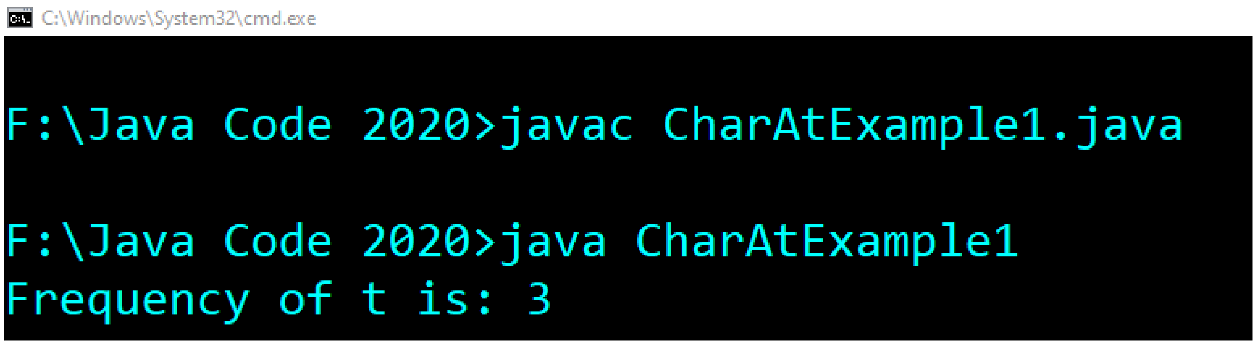
```java
public class CharAtExample {
    public static void main(String[] args) {
        String str = "University of Petroleum and Energy Studies";
        for (int i=0; i<=str.length()-1; i++) {
            if(i%2!=0) {
                System.out.println("Char at "+i+" place "+str.charAt(i));
            }
        }
    }
}
```


C:\Windows\System32\cmd.exe

```
F:\Java Code 2020>java CharAtExample
Char at 1 place n
Char at 3 place v
Char at 5 place r
Char at 7 place i
Char at 9 place y
Char at 11 place o
Char at 13 place
Char at 15 place e
Char at 17 place r
Char at 19 place l
Char at 21 place u
Char at 23 place
Char at 25 place n
Char at 27 place
Char at 29 place n
Char at 31 place r
Char at 33 place y
Char at 35 place S
```

Let's see an example where we are counting frequency of a character in the string.

```
public class CharAtExample1 {
   public static void main(String[] args) {
      String str = "University of Petroleum and Energy Studies";
      int count = 0;
      for (int i=0; i<=str.length()-1; i++) {
         if(str.charAt(i) == 't') {
            count++;
         }
      }
      System.out.println("Frequency of t is: "+count);
   }
}
```

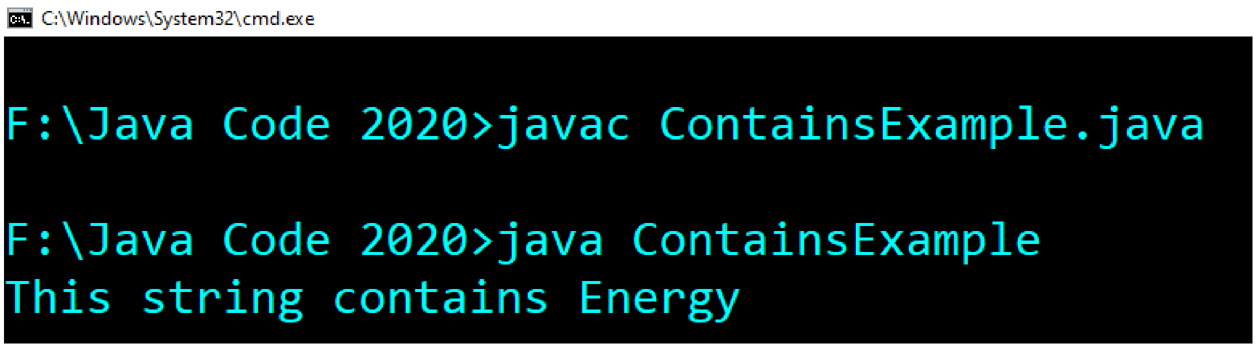


```
C:\Windows\System32\cmd.exe

F:\Java Code 2020>javac CharAtExample1.java

F:\Java Code 2020>java CharAtExample1
Frequency of t is: 3
```

13.Java String contains()

The java string contains() method searches the sequence of characters in this string. It returns true if sequence of char values are found in this string otherwise returns false.

```
public class ContainsExample {
   public static void main(String[] args) {
      String str = "University of Petroleum and Energy Studies";
      if(str.contains("Energy")) {
         System.out.println("This string contains Energy");
      }else
         System.out.println("Result not found");
   }
}
```



C:\Windows\System32\cmd.exe

```
F:\Java Code 2020>javac ContainsExample.java

F:\Java Code 2020>java ContainsExample
This string contains Energy
```

use the following URL for some more string methods:
https://www.tutorialspoint.com/java/java_strings.htm
Assignment 2:
Q 1: What are the advantages and disadvantages of SCP?