Exception Throwable
Error

try(){

}catch(){

}

throw checked

try -> It is a block used to check if the statementsi inside it are generating an exception
catch-> It is block that is used to handle the exception
throw -> Used to generate a new exception
throws -> used for exception routing from current method to the caller method
finally -> a block used to close the resources

Checked
Unchecked

```
getAllEmployees()throws{
select * from employees;
if(result==null)
      throw SQLException;
}
```

```
main(){
try{
getAllEmployees();

}
catch(){
}
}
```

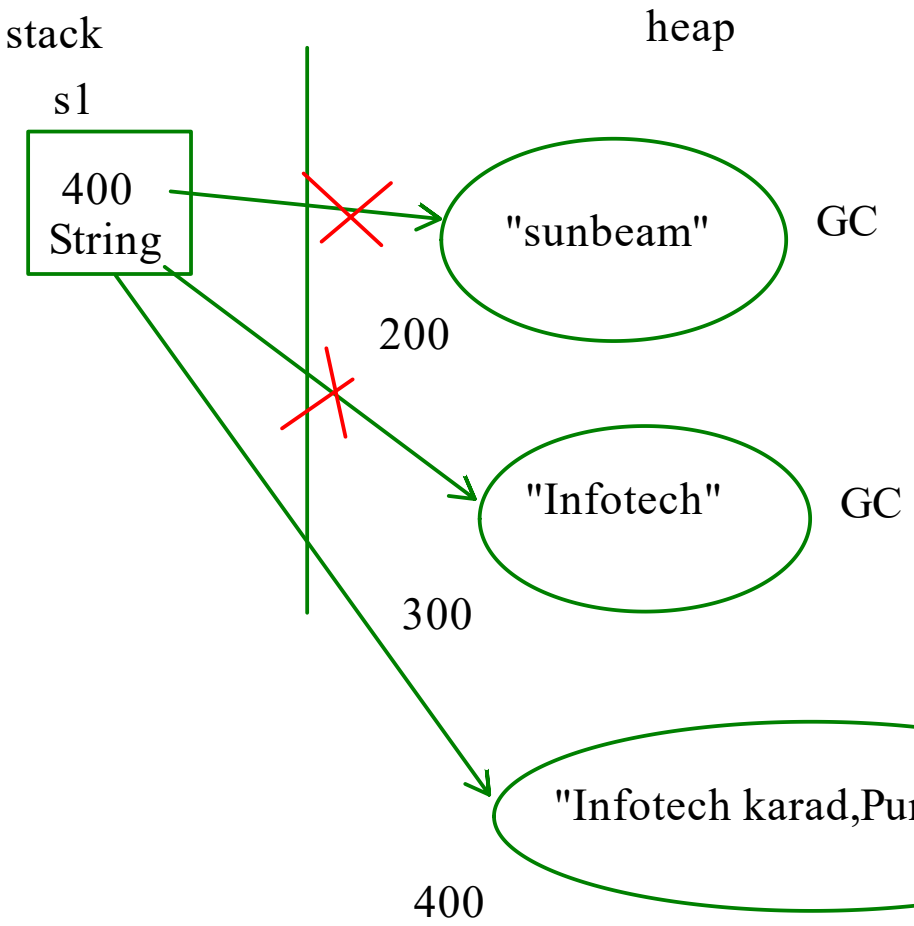
```
main(){
try{
m1();
}
catch(){
}
}
```

```
m1()throws{
m2();
}
```

```
m2() throws checked{
throw checked;
}
```

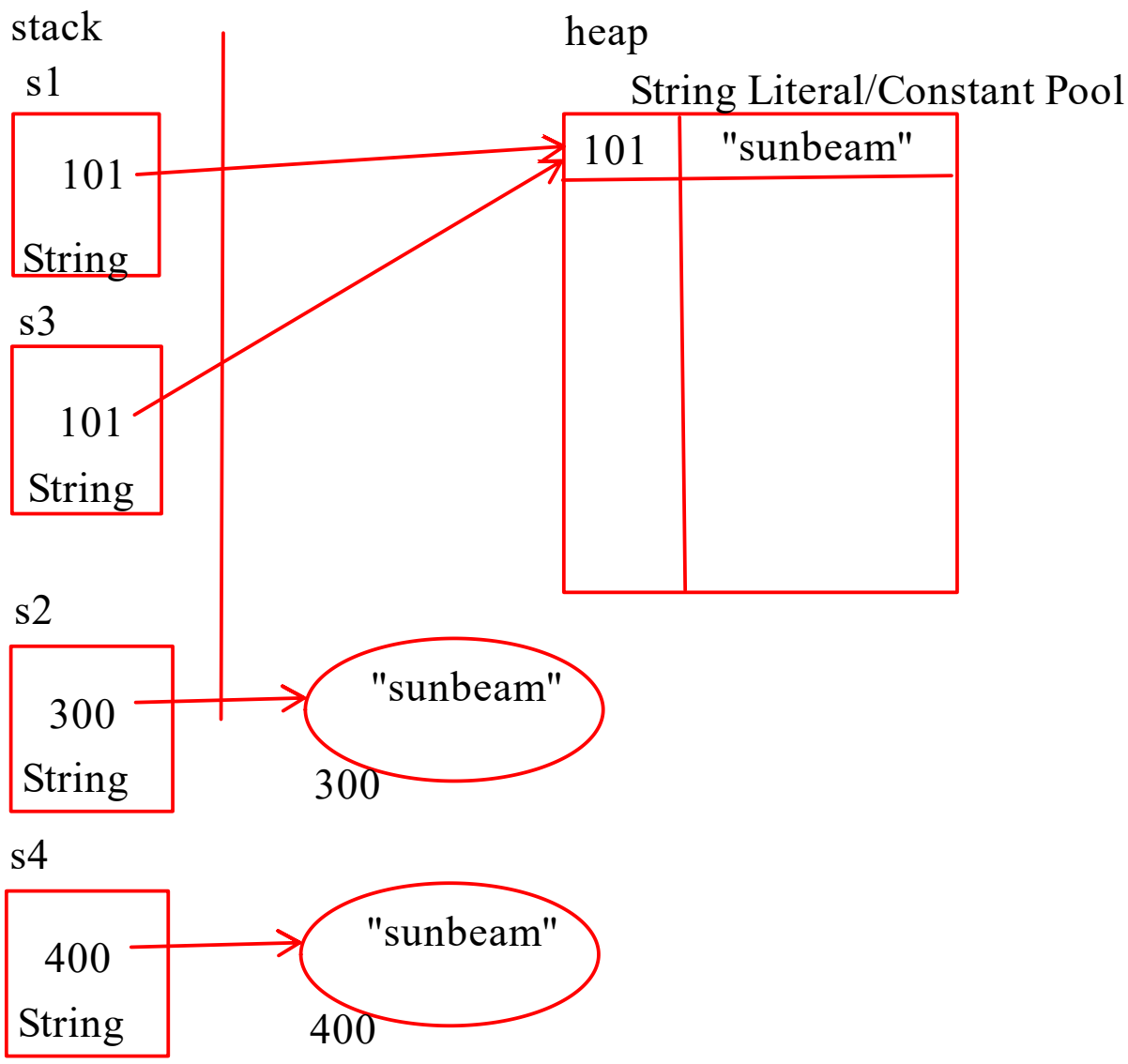
String
- It is a class in java.lang package
- String class is immutable

```
String s1 = "sunbeam";

s1 = "Infotech";

s1 = s1 + "karad,Pune";

String name = "rohan";
```



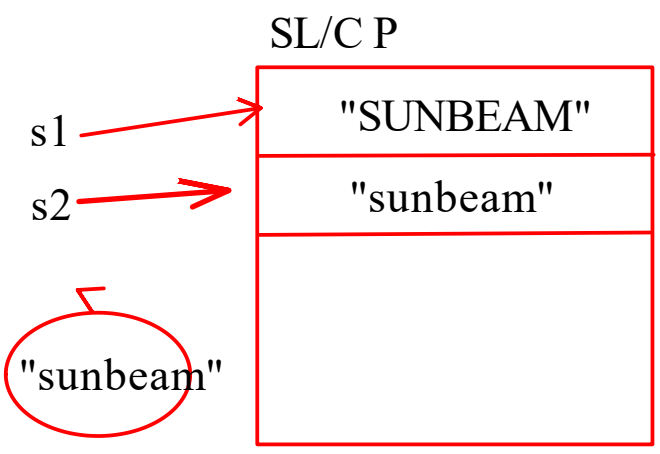
String s1 = "sunbeam"; // String literal
String s2 = new String("sunbeam");

Literals -> Constant values used for initialization
are called as literals



String s3 = "sunbeam"; // String Literal

String s4 = new String("sunbeam");



new String("sunbeam"); "SUNBEAM" //GC

intern();

String
- Oject of String class is immutable
- It is used to work with String literals
- equals and hashcode method from object class are overriden inside String class

String Buffer
- Object of StringBuffer class is mutable
- It cannot work directly with String literals.
- equals and hashcode method from object class are not overriden inside this class
- It is a thread-safe class.

String Builder
- Object of StringBuilder class is mutable
- It cannot work directly with String literals.
- equals and hashcode method from object class are not overriden inside this class
- It is not a thread-safe class.

StringTokenizer
- A class that is used to split the strings into tokens based on the delimeter provided

```
enum ArithmeticOperations{
EXIT,ADD,SUB,MUL
}
```

```
enum ArithmeticOperations{
EXIT,ADD,SUB,MUL
}

 final class ArithmeticOperations extends Enum{
        public static final ArithmeticOperations EXIT;
        public static final ArithmeticOperations ADD;
        public static final ArithmeticOperations SUB;
        public static final ArithmeticOperations MUL;
        private static final ArithmeticOperations[] ENUM$VALUES;

        private ArithmeticOperations(String name, int ordinal){
                super(name,ordinal);
        }

        static{
                EXIT = new ArithmeticOperations("EXIT",0);
                ADD = new ArithmeticOperations("ADD",1);
                SUB = new ArithmeticOperations("SUB",2);
                MUL = new ArithmeticOperations("MUL",3);
                ENUM$VALUES = {EXIT,ADD,SUB,MUL};
        }

        public static ArithmeticOperations[] values(){
                return ENUM$VALUES;
        }

        public static ArithmeticOperations valueof(String s1){
                for(ArithmeticOperations element: ENUM$VALUES)
                        if(element.name().equals(s1))
                                return element;
                return null;
        }
  }
```

Lab Work -
1. String,Buffer,Builder
2. Enum

- Compulsary
- template
- upcasting, downcasting
- interfaces
    - upcasting

```
class Outer{
        class Inner{

        }
}

Outer outer = new Outer();
Outer.Inner i1 = outer.new Inner();
```