Lab-6

1. Demonstrate Various String Constructors with proper Java programs.

2. Demonstrate String length, String literal, String Concat.

```
Public class StringMain{
  Public static void main(String args[]){
   String s= new String(); //without parameters
   String s2= new String("Hello Java");
   System.out.println("s2="+s2);
  ~~Strin~~
   char mychars[] = {'J', 'A', 'V', 'A'};
   String s3 = new String(mychars);
   System.out.println("s3 = + s3);
  String s4 = new String(mychars, 0, 2);
  System.out.println("s4"+s4);
  byte b[]= {65, 66, 67, 68};
  String s5= new String(b);
  System.out.println("s5="+s5);

 // string length, string literal, string concat

  String name= "Santhosh";
  System.out.println("length"= + name.length());
  String branch= "CSE";
  String details = name+branch;
  String details
System.out.println("Concatenated strings ="+ details

}
}
```

```
output :
S2 = Hello java
S3 = java
S4 = ja
S5 = ABCD
Length = 6
Concatenated strings = santhoshCSE
```

(2) WAP to crear an abstract class Bird with methods fly() & make sound () create subclass Eagles and Hawk that Extend class & Implement the respective methods to describe how each bird flies & makes a sound.

```
abstract class Bird {
  abstract void fly();
  abstract void makesound ();

}

class Eagle Extends Bird{
   void fly (){
      System.out.println (" Engle soars high in
          the sky");

   }
   void ~~void~~ make sound (){
   System.out.println (" Eagle makes screeching
                               sound.");

}}
```

Scanned with ACE Scanner

```
class Hawk Extends Bird {
  Void fly(){
    System.out.println ("Hawk glides gracefully
                         in the air");

  }
  Void make sound (){
     system.out.println ("Hawk emits a
     distinctive cry");

class BirdMain {
     Public static Void main (String[] args){
      Eagle e= new Eagle
      e. fly();
      e. make sound ();
      Hawk h = new Hawk ();

      h.fly();
      h. makesound();

    }}
```

```
Eagle soars high in the sky
Eagle makes screeching sound.
Hawk glides gracefully in the air
Hawk emits a distinct cry.
```

16/11/24

③ WAP to create a generic class stack which hold 5 integers and 5 double values

```
Public class GenericStack <T> {
  Private object() stack array;
  Private int top;
  Private static final int max-size = 10;

Public GenericStack(){
   stack Array - new Object[max-size];
    top = -1;

public void push(T element) {
    if (top <max-size -1){
        stack Array [++top] = element;
        System. out. println ("Pushed !" +
      element); }
else {
   System. out. println ("Stack is full. Cannot
    push more elements");

public TPop(){
   If (!isEmpty()){
    @SuppressWarnings ("unchecked")
    T element = (T) stack Array[top--];
   System. out. println ("Popped !" + element);
   return element;
  else{
     System.out.println(" stack is empty.
     Cannot pop element. ");
   return null;
  }}
public  boolean  is empty(){
      return  top ==-1;
      }

Public  stack  void  main (String[]  args){
    GenericStack <Integer> integerStack =
   new  GenericStack<>();
   integer stack. push (1)
   integer stack. push (2)
   integer stack push (3);
   integer stack . pop();

   GenericStack <Double>  doubleStack =
   new  GenericStack <> ();
   double stack. push (1.5);
   double stack push (2.5);

 }}
```

16|1|24

output :

```
 pushed 1
 pushed 2
 pushed 3
 poped 3
pushed 1.5
pushed 2.5
```