16/01/24

Lab 6

1. String Constructors
O/p: ABCDEF

2. Str length
The length of the string is 13

Str literal
Hello World

Str concat
Str1 : Java
Str2 : Programming
Str3 : Java Programming

1.
```
class String
{
public static void main(String args[])
  {
  char c[] {'J', 'a', 'v', 'a'};
  String s1 = new String(c);
  String s2 = new String(s1);
  System.out.println(s1);
  System.out.println(s2);
}
}
```

O/p
JAVA
JAVA

8.
```
public class Main
{
public static void main(String[] args)
{
String test = "testString";
String pattern = "te";
System.out.println(test.startsWith(pattern));
pattern = "est";
System.out.println(test.startsWith(pattern));
}
}
```

Op
true
false

19. abstract Write a java program to create an abstract class Bird with abstract methods fly() and makeSound(). Create subclasses Eagle and Hawk

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

class Eagle extends Bird
{
void fly()
{
System.out.println("Eagle is flying high");
}
void makeSound()
{
System.out.println("Eagle is making a screeching sound");
}
}
```

```
class Hawk extends Bird {
  void fly()
{
System.out.println("Hawk is flying with a
    speed");
}
void makeSound()
 {
  System.out.println("Hawk is making a crying
      sound");
 }
}

public class Main{
 public static void main(String[] args)
  {
  Eagle eagle = new Eagle();
  Hawk hawk = new Hawk();

  eagle.fly();
   eagle.makeSound();
  hawk.fly();
  hawk.makeSound();
 }
}
```

O/p:

Eagle is flying high
Eagle is making a screeching sound
Hawk is flying with a speed
Hawk is making a humming sound

Page No.:
Date:
YOUVA

Q Write a Java Program to create a generic class Stack which holds 5 integers & 5 double values

```
import java.util.EmptyStack;
public class Stack <T>{
  public int maxsize;
  public int top;
  Public Object[] stackArray;
  Public Stack(int size)
   {
     maxSize = size;
     stackArray= new Object[maxSize];
     top = -1;
  }
Public void push(T value)
  {
   if (top < maxSize -1)
    {
     top++;
   stackArray[top]= value;
  }
  else
   {
   throw new RuntimeException("Stack is full");
  }
 }
  public T pop()
    {
    if (! isEmpty())
      {
       return(T) stackArray[top--];
    }
    else
    {
     throw new EmptyStackException();
```

```
public Boolean isEmpty()
 {
  return(top == -1);
}
public int size()
  {
  return top+1;
}
public static void main(String[] args)
  {
Stack<Integer> intStack = new Stack<>(5);
Stack <Double> doubleStack, new Stack <>(5);

for(int i=0; i<5; i++)
  {
  intStack.push(i);
  doubleStack.push((double)i);
}
System.out.println("Integer Stack");
for(int i=0; i<5; i++)
  {
    System.out.println(intStack.pop());
}
  System.out.println("Double Stack");
  for(int i=0;i<5;i++)
    {
    System.out.println(doubleStack.pop());
}
}
}
```

O/p

Int Stack: 4 3 2 1 0

Double Stack: 4.0 3.0 2.0 1.0 0.0

16/1/24