Program1: Creation of class for student information:

```
package sss; import java.util.Scanner;
public class student {
public String vtuno; public String
fullName; public int sem;
public String branchcode;
public String address;
public student(){ Scanner
scanner=new Scanner(System.in);
System.out.print("VTU NO:");
vtuno=scanner.nextLine();
System.out.print("FullName:");
fullName=scanner.nextLine();
System.out.print("Address:");
address=scanner.nextLine();
System.out.print("branch:");
branchcode=scanner.nextLine();
System.out.print("semester:");
sem=scanner.nextInt();}
public void show(){
System.out.println("Entered Data");
System.out.println("VTU No.
"+vtuno);
System.out.println("FullName:"+fullName); System.out.println("sem:"+sem);
System.out.println("Branch:"+branchcode);System.out.println("Address:"+address);}public static void main(String[]
args){student std=new
student();std.show();}}
```

Output:

```
VTU NO:3pd15cs092
FullName:xyz Address:kalaburagi
branch:cse semester:6 Entered Data
VTU No. 3pd15cs092 FullName:xyz
sem:6 Branch:cse Address:Kalaburagi
package sss;
```

Program2: String Handling

```
import java.util.Scanner;
public class Stringops { public static
void main(String[] args) {
            StringBuffer s=new
StringBuffer(); String str2;
Scanner input=new
Scanner(System.in);
s.append("java programming  ");
System.out.println("the current
capacity of StringBuffer "
+s.capacity());
System.out.println("initial String
without append: " +s);
System.out.println("enter a String to
append:  ");
str2=input.nextLine(); s.append(""
+str2); System.out.println("initial
String after append: " +s); s.reverse();
System.out.println("initial String after
reversing: " +s); int i=0; do {
s.replace(i, i+1,
s.substring(i,i+1).toUpperCase());
i=i+1; } while(i>0 && i<s.length());
System.out.println("after
reversing:"+s); }}
```

Output:

```
the current capacity of StringBuffer 34
initial String without append: java
programming
enter a String to append:  lab
initial String after append: java
programming  lab
initial String after reversing: bal
gnimmargorp avaj
after reversing:BAL
GNIMMARGORP AVAJ
```

Program3a: Constructor Overloading

```
package sss;
import java.util.Scanner;
class prog5 { int a,b;
Scanner s1=new Scanner(System.in);
prog5() { System.out.println(“Enter
any 2 integers:”); a=s1.nextInt();
b=s1.nextInt(); } void display() {
System.out.println(“Addition=”+(a+b)
);
S.o.p(“Subtraction=”+(a-b));
S.o.p(“Multiplication=”+(a*b));
S.o.p(“Division=”+(a/b)); }
prog5(float a1, float b1) {
S.o.p(“Addition=”+(a1+b1));
S.o.p(“Subtraction=”+(a1-b1));
S.o.p(“Multiplication=”+(a1*b1));
S.o.p(“Division=”+(a1/b1)); } void
display(int x) { S.o.p(“Square of” +x+
“is”+(x*x)); } } class main { public
static void main (String [] args) {
Scanner s1=new Scanner(System.in);
S.o.p(“Arithmetic Operation on
integer”); prog5 a=new prog5();
a.display(); S.o.p(“\n Arithmetic
Operation on float”); S.o.p(“Enter any
two float numbers”); float
a1=s1.nextFloat();float
a2=s1.nextFloat(); prog5 arth1=new
prog5(a1,a2); S.o.p(“\n Enter number
to find sqaure”); int x=s1.nextInt();
a.display(x); } }
```

Output:

```
Enter any two integers 29 20
Addition=49 Subtraction=9 Multi=580
Division=1
```

Program3b: To implement inner class

```
package accessprotection; import
java.io.*; class  outer { int  outdata =
10;
void  display(){ inner  inobj = new
inner(); System.out.println("Accessing
from outer class");
System.out.println("The value of
outdata is " +outdata);
System.out.println("The value of
indata is "   +inobj.indata); } class
inner { int  indata = 40; void
inmethod() {
System.out.println("Accessing from
inner class"); System.out.println("The
sum of indata & outdata is " +(outdata
+ indata)); }}} class  AccessMain{
public  static  void  main(String
args[])
{outer  outobj  = new  outer();
outobj.display();
outer.inner  inobj1 = outobj.new
inner();
inobj1.inmethod(); } }
```

Output:

```
Accessing from outer class
The value of outdata is 10
The value of indata is 40
Accessing from inner class
The sum of indata & outdata is 50
```

Program4a: Single and Multilevel Inheritance

```
package sss; class sup { int x;  sup(int
x)
{ this.x=x; } void display() {
System.out.println("sup+x = " +x); } }
class supr extends sup { int y;
supr(int x,int y) { super(x); this.y=y; }
void display() {
System.out.println("sup-x = "+x);
System.out.println("sup-y = "+y); } }
class prog4a {
public static void main(String args[]) {
supr s1 = new supr(100,200);
s1.display(); } }
```

Output: sup-x = 100 sup-y = 200

Program4b: Method Overriding

```
package sss; class parent {  void
show()
{ System.out.println("Parents
show()"); } }  class child extends
parent {
@Override void show() {
System.out.println("Child Show()"); }
}
public class overriding { public static
void man(String [] args) { parent
obj1=new parent(); obj1.show();
parent obj2=new child(); obj2.show();
}}
```

Output: sup-x = 100 sup-y = 200

Program5: Multiple inheritance using interfaces

```
package sss; interface compute {
double calculate(); }
class Rectangle implements compute {
double l,b; void getvalues() { l =
10.5F;
b=7.3F; } public double calculate() {
return (l*b);  } }
class TRI extends Rectangle
implements compute { public double
calculate() {
return (0.5*b*l); } } class prog4b {
public static void main(String[] args)
{ Rectangle R = new Rectangle();
 R.getvalues();
 System.out.println("Area of Rectangle
= " + R.calculate()); TRI T =new
TRI();
T.getvalues();
System.out.println("Area of triangle =
" + T.calculate()); }}
```

Output:

```
Area of Rectangle =
76.65000200271606
Area of triangle = 38.32500100135803
```

Program7: Program to create account with balance……….

```
package sss; import java.util.*;
class Account { int curBalance,amt;
public Account(){ curBalance=500; }
void Deposit() {
Scanner s=new Scanner(System.in);
System.out.println("\nEnter the
amount:");amt=s.nextInt();
curBalance+=amt;System.out.println("
\n Balance:" +curBalance); }
void withdraw() { Scanner s=new
Scanner(System.in);
System.out.println("\n Enter the
amount"); amt=s.nextInt(); try {
if((curBalance-amt)<500) throw new
LessBalanceException(amt);
curBalance=amt;
System.out.println("\nBalance left:
"+curBalance); }
catch (LessBalanceException le) {
System.out.println(le); } }
void checkBal() {
System.out.println("\nBalance in your
a/c: "+curBalance); } }
class LessBalanceException extends
Exception { int am; public
LessBalanceException(int x) { am=x;
System.out.println("Less Balance :
"+x); } public String toString() {
return("You cannot with draw the
amount,less balance"+am+ "Now"); }
}
class New_class { public static void
main(String[] ar) { int ch;
Scanner s=new Scanner(System.in);
Account a=new Account();
while(true) {
System.out.println("1:Deposit\t2:With
drawl\t3:Balance\t4:Exit\n");
System.out.println("Enter your
choice:"); ch=s.nextInt();
switch(ch) {
case 1: a.Deposit(); break;
case 2:   a.withdraw(); break;
case 3: a.checkBal(); break;
case 4: return;
default: System.out.println("Invalid
choice\n"); return;} } } }
```

Program8: Multi Threading

```
package demotest;
public class thread_example1
implements Runnable {
@Override  public void run() { }
public static void main(String[] args) {
Thread thread1 = new Thread();
thread1.start(); try {
thread1.sleep(1000);
} catch (InterruptedException e) {
e.printStackTrace(); }
thread1.setPriority(1);
int priority = thread1.getPriority();
System.out.println(priority);
System.out.println("Thread Running");
}}
```

Output:

```
5
Thread Running
```

Program6: Polymorphism through method overloading

```
package sss; class Pattern { public
void display() { for (int i = 0; i < 10;
i++) { System.out.print("*"); } }
public void display(char symbol) { for
(int i = 0; i < 10; i++) {
System.out.print(symbol); }}}
class Main { public static void
main(String[] args) { Pattern d1 = new
Pattern();
d1.display();System.out.println ("\n");
d1.display('#'); }}
```

Output:

```
**********
##########
```

Program9: Producer Consumer Test

```
class Consumer extends Thread {
private Sync sync; private int number;
public Consumer(Sync c,int number) {
sync=c;
this.number=number; } public void
run()  { int value=0; for(int
i=0;i<5;i++)  { value=sync.get();
System.out.println("Consumer#"+this.
number +"got:" +value); try{
sleep((int)(Math.random()*10000));}
catch(InterruptedException e){} }}}
class Producer extends Thread{ private
Sync sync; private int number; public
Producer(Sync c,int number){ sync=c;
this.number=number; } public void
run(){ for(int i=0;i<5;i++){
sync.put(i);
System.out.println("Produces"+this.nu
mber+"put:"+i); try{
sleep((int)(Math.random()*10000));
}catch(InterruptedException e){} } } }
class ProducerConsumerTest{ public
static void main(String[] args) { Sync
c=new Sync(); Producer p1=new
Producer(c,1); Consumer c1=new
Consumer(c,1); p1.start(); c1.start();
} } class Sync{ private int contents;
private boolean available=false;
public synchronized int get() {
while(available==false) { try { wait();
} catch(InterruptedException e){} }
available=false; notifyAll(); return
contents; } public synchronized void
put(int value) {
while(available==true){
try{ wait(); }
catch(InterruptedException e){} }
contents=value; available=true;
notifyAll(); } }
```

Output: Consumer#1got:0 **Produces 1put:0** Consumer#1got:1 **Produces 1put:1**

Program10: Applet

```
import java.awt.*; import
java.awt.event.*; import java.applet.*;
public class ap extends Applet
implements ActionListener { Label
label1, label2, label3; TextField tf1,
tf2, tf3; Button b1, b2; String
whichButtonClk; public void init(){
System.out.println("Initializing an
applet"); label1 = new
Label("Number1"); tf1= new
TextField(10); label2 = new
Label("Number2"); tf2= new
TextField(10); b1 = new
Button("Add");
b2= new Button("Subtract");
add(label1); add(tf1); add(label2);
add(tf2); add(b1); add(b2);
tf1.addActionListener(this);
tf2.addActionListener(this);
b1.addActionListener(this);
b2.addActionListener(this); } public
void actionPerformed(ActionEvent
ae){
if(ae.getActionCommand().equals("Ad
d") ||
ae.getActionCommand().equals("Subtr
act")) {
whichButtonClk=ae.getActionComma
nd(); repaint(); } }
public void paint(Graphics g) {
g.drawString("Please enter two
numbers to perform math operations",
10,130);
if(tf1.getText().equals("") &&
tf2.getText().equals("")) { } else {
Integer i1= new Integer(tf1.getText());
Integer i2= new Integer(tf2.getText());
int sum = i1+i2; int subtract=i1-i2;
if(whichButtonClk.equals("Add"))
g.drawString("Your sum is "+ sum,
10,190);
if(whichButtonClk.equals("Subtract"))
g.drawString("Your subtract is "+
subtract, 10,190); } } }
html
<applet code=applet.class width=500
height=500>
</applet>
```