Java Programs:

Program 1:

```
// Polymorphism Dynamic method dispatch


class Account{
int amount;


Account(){}
Account(int amount){
this.amount=amount;
}


void calculateInterest(){
        double intrst=(amount*10)/100;
System.out.println("Account"+amount);
System.out.println("Account intrst:"+intrst);
}
}


class SavingsAccount extends Account{



SavingsAccount(int amount){
this.amount=amount;
}
```

```
void calculateInterest(){
double intrst=(amount*10)/100;
System.out.println("SavingsAccount"+amount);
System.out.println("SavingssAccount intrst:"+intrst);
}
}

class LoansAccount extends Account{

LoansAccount(int amount){
this.amount=amount;
}

final void calculateInterest(){
double intrst=(amount*14)/100;
System.out.println("LoansAccount"+amount);
System.out.println("LoansAccount intrst:"+intrst);
}
}

class AccountDemo{
        public static void main(String args[]){
                Account ac;
                LoansAccount la=new LoansAccount(1000);
```

```
                ac=la;
                ac.calculateInterest();
                SavingsAccount sa1=new SavingsAccount(5000);
                ac=sa1;
                ac.calculateInterest();
                SavingsAccount sa2=new SavingsAccount(500000);
                ac=sa2;
                ac.calculateInterest();

        }

}
```

Program 2:

```
// polymorphism example 2
class Animal{

Animal shout(){

        System.out.println("I can shout");
        return new Animal();
}

}
```

```
class Dog extends Animal{
Dog shout(){
		System.out.println("I can shout like bow bow bow");
	return new Dog();
	}

}

class Cat extends Dog{

Cat shout(){
		System.out.println("I can shout like meaw meaw meaw");
	return new Cat();
	}
}
class AnimalDemo{
	public static void main(String args[]){
		Animal a;
		Dog d1=new Dog();
		a=d1;
		a.shout();
		a=new Cat();
		a.shout();
	}
}
```

Program 3:

```
// Final Key word
class FinalA{
final int a;

FinalA(){a=10;}
FinalA(int a){this.a=a;}
final int method1(int a,int c){
        final int b=a;
System.out.println("FinalA method1"+b);
return a;
}
final void method1(){

System.out.println("FinalA method1 no argument");

}
}
final class FinalB extends FinalA{

void method2(int a,int b){
System.out.println("FinalB method1");



}
```

```
}
class FinalDemo{
public static void main(String args[]){
FinalA a=new FinalA();
a.method1();
}
}
```

Program 4:

```
class Date{
int dd,mm,yy;
void setDate(int dd,int mm,int yy){
this.dd=dd;
this.mm=mm;
this.yy=yy;
}
void showDate(){
System.out.println("The date is:"+dd+"-"+mm+"-"+yy);
}
}

class DateTime extends Date{
int hh,min;
void setDateTime(int hh,int min){
```

```
this.hh=hh;
this.min=min;
setDate(3,7,24);
}

void showDateTime(){
System.out.println("The Date and Time is:"+dd+"-"+mm+"-"+yy+" "+hh+":"+min);
}
}

class DateTimeDemo{

	public static void main(String args[]){

		DateTime dt=new DateTime();
		dt.setDateTime(04,26);
		dt.showDateTime();

		Date d=new Date();
		d.setDateTime(04,26);

	}
}
```

Program 5:

```
// super keyword
class One{
        int i;
        One(){}
One(int i){
        this.i=i;
System.out.println("One"+i);
}

}
class Two extends One{
int i;
Two(int i,int j){
super(j);
        this.i=i;
        System.out.println("Two"+super.i);

}

}
class SuperDemo{
public static void main(String args[]){
Two t=new Two(20,30);
//t.show();
```

```
}
}
```

Program 6:

```
//Usage of this and super
class SuperClass{
int i;
SuperClass(){}
SuperClass(int i){
this.i=i;
System.out.println("Super class"+i);
}

}

class ChildClass extends SuperClass{
        int i;
ChildClass(int i,int j){
        super(j);
        this.i=i;
System.out.println("Child class"+i);
System.out.println("super class variable"+super.i);
```

```
}
}

class SuperClassDemo{
public static void main(String args[]){
ChildClass cc=new ChildClass(10,20);
}
}
```

Program 7:

```
//inheritance
class Teachers{
int age;
String name;
double salary;
Teachers(int age,String name,double salary){
this.name=name;
this.salary=salary;
this.age=age;
}
String getName(){
return name;
}
double getSalary(){
return salary;
}
```

```
int getAge(){
return age;
}

}
class Admin extends Teachers{
Admin(int age, String name, double salary){
super(age,name,salary);
}
void display(Teachers t){
System.out.println(t.salary);
}
}

class MainDemo{
public static void main(String args[]){
Teachers t1=new Teachers(30,"A",30000);
Admin a1=new Admin(30,"admin1",50000);
a1.display(t1);
}
}
```

Program 8:

```
//Overloading Example
```

```
class Overloaded{

void add(int a,int b){
System.out.println("add(int a, int b):"+(a+b));
}

double add(double b,int a){
System.out.println("int add(int a,int b):"+(a+b));
return (a+b);
}

void add(int a,double b){
System.out.println("add(int a,double b):"+(a+b));
}
}
class OverloadedDemo{
        public static void main(String args[]){
                Overloaded o = new Overloaded();
                o.add(10,20);
                o.add(10.5,20);
                o.add(3,4.5);

        }

}
```

Program 9:

```
//Static Block Example


class StaticBlock{


        int a;


        void setA(int a){
    this.a=a;


        }
        int getA(){
                return a;
        }


}
class StaticBlockDemo{
        static{
        StaticBlock obj2=new StaticBlock();
                obj2.setA(30);
                int tmp=obj2.getA();
                System.out.println("Static block value:"+tmp);
        }
        public static void main(String args[]){
                StaticBlock obj1=new StaticBlock();
```

```
                obj1.setA(10);
                int tmp=obj1.getA();
                System.out.println("value:"+tmp);


        }
}
```