Codes for constructor

1.
```
class Student3{
int id;
String name;

void display()
{
System.out.println(id,name);
}
public static void main(String args[]){
Student3 s1=new Student3();
Student3 s2=new Student3();
s1.display();
s2.display();
}
   }
```

2.
```
class Student5{
int id;
String name;
int age;
Student5(int i.String n){
id = i;
name = n;
}
Student5(int i,String n,int a){
id = i;
name = n;
age=a;
}
void display(){System.out.println(id+" "+name+" "+age);}

public static void main(String args[]){
Student5 s1 = new Student5(111,"Karan");
Student5 s2 = new Student5(222,"Aryan",25);
s1.display();
s2.display();
}
}
```

3.
```
public class Cube1 {

int length;
int breadth;
int height;
public int getVolume() {
return (length * breadth * height);
}
Cube1() {
length = 10;
breadth = 10;
height = 10;
}
Cube1(int l, int b, int h) {
length = l;
breadth = b;
height = h;
}
public static void main(String[] args) {
Cube1 cubeObj1+cubeObj2;
cubeObj1 = new Cube1();
cubeObj2 = new Cube1(10, 20, 30);
System.out.println("Volume of Cube1 is : " +
cubeObj1.getVolume());
System.out.println("Volume of Cube1 is : " +
cubeObj2.getVolume());
}
}
```

4.
```
public class ConstructorDemo{
   public ConstructorDemo(){
      System.out.println("Inside no argument constructor");
   }

   public Constructor(String name){
      System.out.println("Inside one argument constructor in Java
with name: " + name);
   }
```

```
    public static void main(String args[]) throws IOException {

      ConstructorDemo d = new ConstructorDemo();
      ConstructorDemo e new ConstructorDemo("Testing");
}
}
```

5.
```
class Candid
{
String name;

int age;

Student(String n,int a)

{

name=n;

age=a;

}

class Student extends Candid

{

int id;

Student(String n,int a,int i)

{

super(n,a);

id=i;

}

}

class Faculty extends Candid

{
```

```
int ph;

Faculty(String n,int a,int ph)

{

Super(n,a);

ph=p;

}

}
```

6.
```
class Dog extends Animal {
String name;
String breed;


public Dog(s) {
name = s;
breed = "unknown";
}


public Dog(String name, String breed) {
this.name = name;
this.breed = breed;
}
}
```

7.
```
public class Platypus {

String name;

Platypus(String ) {

name = input;

}
```

```
Platypus() {

this("John/Mary Doe");

}

public static void main(String args[]) {

Platypus p1 = new Platypus("digger");

Platypus p2 = new Platypus();

}

}
```

8.
```
class Mammal {

void getBirthInfo() {

System.out.println("born alive.");

}

}

class Platypus extends mammal {

void getBirthInfo() {

System.out.println("hatch from eggs");

System.out.print("a mammal normally is ");

super.getBirthInfo();
```

```
}

}
```

9.
```
package beginnersbook.com;
public class ConstOverloading
{
private int rollNum;
ConstOverloading()
{
rollNum =100;
}
ConstOverloading(int,rnum)
{
this();
rollNum = rollNum+ rnum;
}
public int getRollNum() {
return rollNum;
}
public void setRollNum(int rollNum) {
this.rollNum = rollNum;
}
}
class TestDemo{
public static void main(String args[])
{
ConstOverloading obj = new ConstOverloading(12);
System.out.println(obj.getRollNum());
}
}
```

10.
```
Class enumCar {
lamborghini(900),tata(2),audi(50),fiat(15),honda(12);
private int price;
Car(int p) {
price = p;
}
int getPrice() {
return price;
}
}
public class Main {
public static void main(String args[]){
System.out.println("All car prices:");
for (Car c : Car.values)
System.out.println(c + " costs "
+ c.getPrice() + " thousand dollars.");
}
}
```

11.
```
public class ConstructorChaining{

    public static void main(String args[]) {

            Derived.sub = new Derived("Test");
    }
}

class Base{
    protected String name;

    public Base(){
        this("");
        System.out.println("Inside no argument constructor of
Base class");
    }

    public Base(String name){
        this.name = name;
        System.out.println("One arg constructor of Base class");
```

```
    }
}

class Derived extends Base{

   public Derived(){
       System.err.println("Inside no argument constructor of
Derived class");
   }

   public Derived(String name){
       super(name);
System.out.println("Inside one arg constructor from Derived
class");
   }
}
```

12.
```
class Bicycle
{
private int gear;
private int cadence;
private int speed;
public class(int startCadence, int startSpeed, int startGear)
{
gear = startGear;
cadence = startCadence;
speed = startSpeed;
}

public void ShiftUp() {
gear = gear + 1;
}
}
```

13.
```
public class Person {
```

```
private String name;

public Person(nameParameter ) {
this.name = nameParameter;
}
public String toString() {
return "My name is : " + this.name;
}

public static void main( String [] args )  {

Person one = new Person("Patrick");
System.out.println( one );
Person two = new Person("Oscar");
System.out.println( two );
}
}
```

14.
```
class Rectangle {
int length;
int breadth;

Rectangle(int len,int bre)
{
length  = len;
breadth = bre;
}
}

RectangleDemo {
public static void main(String args[]) {

Rectangle r1 = new Rectangle(20,10);

System.out.println("Length of Rectangle : " + r1.length);
System.out.println("Breadth of Rectangle : " + r1.breadth);

}
}
```

15.
```
package com.howtoprogramwithjava.constructors;

public class HumanBeingProgram extends Animal
```

```
{
public static void main (String[] args)
{
HumanBeing me = new HumanBeing:
output(me);

HumanBeing you = new HumanBeing("blue", "female", "Jane Doe");
output(you);
}

private static void output(HumanBeing human)
{
System.out.println(human.getName() + "'s eyes are: " +
human.getEyeColor());
System.out.println(human.getName() + " is " + human.getSex());
System.out.println("--------------");
}
}
```

16.
```
public class Department {

private name;
private int id;

public String getName() {
return name;
}

public void setName(String name) {
this.name = name;
}

public int getId() {
return id;
}

public void setId(int id) {
this.id = id;
}

public Department(int id, String name) {
this.id = id;
```

```
this.name = name;
}

public Department(Department oldDepartment) {
this.id = oldDepartment.id;
this.name = oldDepartment.name;
}


public String toString() {
return "Department Id: " + id + "\tDepartment Name: " + name;
}
}
```

17.
```
public class Employee {

private String name;
private int id;
private Department department;

public Department getDepartment() {
return department;
}

public String getName() {
return name;
}

public int getId() {
return id;
}

public void setDepartment(Department department) {
this.department = department;
}

public void setName(String name) {
this.name = name;
}

public void setId(int id) {
this.id = id;
}

public Employee(int id, String name, Department department) {
```

```
this.id = id;
this.name = name;
this.department = department;
}

public Employee(Employee oldEmployee) {
this.id = oldEmployee.id;
this.name = oldEmployee.name;
this.department = oldEmployee.department;
}

public String toString()
{
return "Employee Id: " + id + "\tEmployee Name: " + name + "\t"
+ department;
}
}
```

18.
```
enum Apple {
  A(10), B(9), C(12), D(15), E(8);

  private int price;   Apple(int p)
{

    price = p;
  }

  int getPrice() {
    return price;
  }
}

public class EnumDemo3 {
  public static void main(String args[]) {
    Apple ap;
```

```
    System.out.println(Apple.A.getPrice());

    System.out.println("All apple prices:");
    for (Apple a : Apple.values)
   System.out.println(a + " costs " + a.getPrice() + " cents.");
  }
}
```

19.
```
class Volume {

   public void findVolume ( int s) {
   System.out.println ( "Volume of cube is "+ ( s * s * s ) );
   }

   public void findVolume ( int r, int h )
{
System.out.println ( "Volume of cylinder is "+ ( 3.14 * r * r *
h) );
   }

   public void findVolume ( int l, int b, int h)
{
   System.out.println ("Volume of cuboid is " + ( l * b * h ) );
   }
}
class VolumeTest {

   public static void main(String[] args) {
      Volume v=new Volume();
      v.findVolume(3);
      v.findVolume(3,4);
      v.findVolume(3,4,7);
   }
}
```

20.
```
package createobject;
```

```
import java.awt.Color;

public class Rectangle   {
private String name;
private int length;
private int width;
private Color color;
public Rectangle {
this("undefined", 10, 10, Color.white);
}
public String toString() {
return " Name: " +
name+" width: " + width + " length: " + length + " color: " +
color;
}
public Rectangle(String name, int width, int length, Color
color)
{
this.name=name;
this.length = length;
this.width = width;
this.color = color;
}
public String getName() {
return name;
}
public void setName(String name) {
this.name = name;
}
public int getLength() {
return length;
}
public void setLength(int length) {
this.length = length;
}
public int getWidth() {
return width;
}
public void setWidth(int width) {
```

```
this.width = width;
}
public Color getColor() {
return color;
}
public void setColor(Color color) {
this.color = color;
}
}
```