Output :-

Area = 200 m² // SOP in PSVM

Volume = 6000 m³

Date

Expt. No. Page No.

⇒ WAP to show working of inheritance.

```
class Room
{
 int l;
 int b;

 Room( int x, int y)
    {
     l = x;
     b = y;
    }
 int area ()
 return (l * b);
  }

class BedRoom () extends Room
   {
    int height;
BedRoom (int x, int y, int z)
     {
       super(x,y);
       height = z;
     }
     int volume ()
      {
      return (l * b * h)
      }
     }
```

Teacher's Signature

Expt. No. _____ Page No. _____

```
class innt()
{
Public static void main(String args[])
{
BedRoom B= new BedRoom (10,20,30);

    int area = B.area();
    int volume = B.volume();

System.out.print (area);
system.out.println(volume);
  }
  }
```

Teacher's Signature _____

Output :-

```
Output is : 300
```

Expt. No. Page No.

⇒ WAP to show working of constructor.

```
class BOX {
  double width;
  double height;
  double depth;
   }
BOX( double w, double h, double d)
   {
   width = w;
   height = h;
   depth = d;
   }
double volume()
   {
  return (width x height x depth)
   }
class BOXDEMO
   {
  PSVM
   {
  Box mybox1 = newbox(10, 20, 15);
   vol =  mybox. volume();
   }
   }
```

Output

NO parameters
a= 10
a=10 , b = 20
result = 123.25

Expt. No. _____ Page No. _____

Q7 WAP to show working of overloading

```
class OverloadDemo
  {
void test()
{
System.out.println("No parameters");
  }
void test (int a)
   {
  system.out.println("a");
    }
  void test (int a, int b)
   {
  system.out.println("a,b")
     }
 Double test (double a)
{
system.out.println("double a")
 return (a*a);
   } }
class overload {
 psvm
OverloadDemo ob= new OverloadDemo();
 double result();
  {
  ob.test()
  ob.test(10);
  ob.test(20,20);
result = ob.test(123.25);
System.out.println("result=", result);
   } } }
```

Teacher's Signature _____

Output:-

3 //calling the overriding function in B class hence print the value of C

Date

Expt. No. Page No.

-D WAP to showcase working of overriding

```
class A
 {
int (i, j);
A (int a, int b)
  {
   i=a;
   j=b;
  }
  void show()
  {
system.out.print. (i, j)   }}
class D extends A {
  int k;
 b(int a, int b, int c)
 {
 super (a, b);
   k=c;  }
void show ()
{
system.out.println(K); }}
class override
 {
PSVM {
B subb =   new B (1, 2, 3);
subob. show()
  }
  }
```

Teacher's Signature

Output:-

inside A's call-me method // r=a
r.callme()

inside B's call-me method // r=b
r.callme()

inside C's callme method r=c
r.callme()

Date

Expt. No. _____ Page No. _____

=> WAP to show working of dynamic method dispatch.

```
class A
 {
void callme
{
S.O.P (inside A's call-me method)
}}
class B extends A
 {
 void callme
  {
 S.O.P (insides B's call me method) }}
 class C extends A
  {
  void callme
   {
 S.O.P (insides C's callme method)
  }
  }

class Dispatch {
  PSVM {
  A a = new A();
  B b = new B();
  C c = new C();
    A r;
  r = a;
  r.callme();
  r = b;
  r.callme();
  r = c
  r.callme() }}
```

Teacher's Signature _____

Output:-

B is implementation of callme // b.callme()

This is a concrete method // b.callmetoo()

Expt. No. _____ Date _____

Page No. _____

=> WAP to show working of Abstract class

```
abstract class A
{
abstract void callme();
void callme too()
{
SOP ("This is a concrete method")
}}
class B extends A
{
  void callme()
{
SOP ("B is implementation of callme");
}}
class AbstractDemo
  {
  PSVM
  {
  B b = new B();
    b.callme();
    b.callmetoo();
    }
    }
```

Output :-

```
Caught inside DemoProc          //try in PSVM
Recaught Demo                   //SOP
```

Date

Expt. No. Page No.

⇒ WAP to show working of try-catch using throw

```
class ThrowDemo
{
static void DemoProc()
{
try {
throw (new NullpointerException("demo"));
}
catch (NullpointerException e)
{
SOP("Caught inside DemoProc");
throw e;
}}
PSVM
{
try{
demoProc();
}
catch (NullpointerException e)
}
S.O.P("Recaught" + e);
}}
```

Teacher's Signature

Output :-

Inside throwOne // PSVM

Date

Expt. No. Page No.

⇒ WAP to show working of trycatch using Throws.

```
class ThrowsDemo
 {
static void throwone () throws IllegealaccessException
{
SOP("Inside throw one");
Throw new IllegealAccessException ("demo");
 }
public static void main (string args[])
 {
   throwone();
  }
  }
```

Teacher's Signature

Output :-

```
ProcA is finally                    //finally
exception caught                    // try
Inside Proc B                       // Proc B in PSVM
Inside Proc C                       // Proc C in PSVM
```

Date

Expt. No.

Page No.

Q WAP to show working of try-catch using finally.

```
class FinallyDemo
{
static void procA()
{
try {
S.O.P("Inside ProcA");
throw new RunTimeException("Demo");
}
Finally {
S.O.P("ProcA is finally");
}}
static void ProcB()
{
try{
SOP("Inside Proc B");
finally{
S.O.P (Inside Proc B)
}
}
Static Void Proc C()
{
SOP("Inside Proc C")
}}
PSVM
{
try{
```

Teacher's Signature

Expt. No.

Page No.

```
Proc A();
  }
catch (Exception e)
  {
SOP ("exception caught");
  }
Proc B();
Proc C();
  }
  }
```

Output:-

My exception a= 1
caught a

Normal exit

// a>10

Date

Expt. No. Page No.

Java Build-In-Exception

WAP to create your own exception

```
class MyException extends Exception
 {
private int detail;
  MyException(int a)
   {
   detail = a;
   }
  Public String to string ()
   {
  return "myException ("+detail+")";
       }
       }
class ExceptionDemo
 {
Static Void compute (int a) throws MyException
  {
SOP ("called compute ("+a")");
  if a>10
  throw new myException (a);
  SOP ("Normal Exit");
   }
PSVM
   {
   try
    {
```

Teacher's Signature

```
compute (1);
compute (20);
}
catch (MyException e)
{
SOP("Caught" + e);
}
}
```

Output :-

Main Thread
Current Thread

Main Thread
current thread

Main Thread
current Thread

Main Thread
current thread

Main Thread
current Thread.

Date

Expt. No. _____ Page No. _____

⇒ WAP to show the working of Multithreading.

```
class CurrentThreadDemo
  {
PSVM
Thread t = Thread.currentThread();
SOP ("current Thread" +t);
 t.setName ("mythread");
 SOP ("After name change" +t);
   try
   {
 for (int n=5; n>0; n--)
     {
     SOP (n);
  Thread.sleep(1000)
    }
    }
  catch (InterruptException e)
    {
  SOP( "main thread interrupted")
     }
     }
     }
```

Teacher's Signature _____