Sanjay Kumar Say
19231115J

Assignment - 2

Inheritance

> Single
> multilevel
> multiple (not supported in Java. use interface for it)
> hairarchical
> hyboid

Single inheritance

```
Class A {
    void disA () {
System.out.println ("A"); }}
Class B extends A {
    void disB() {
System.out.println ("B"); }}
Class Test {
    Public static void main (String [] args ) {
        B obj = new B();
        obj.disA ();
        obj.disB (); }}
```

output:-

A
B

Multilevel

```
Class A {
    void disA () {
System.out.println ("A"); }}
```

```
class B extends A{
    void disB() {
System.out.println("B");}}
class C extends B {
    void disC() {
System.out.println("C");}}
class Main {
 Public static void main (String[] args ){
        C d = new C();
         d.disA();
         d.disB();
         d.disC();}}
```

output

A
B
C

heirarchy

```
class A {
    void disA() { System.out.println("A");}}
class B extends A {
    void disB() { System.out.println("B");}}
class C extends A {
    void disB() { System.out.println("C");}}
class main {
    Public static void main (String[] args) {
        C d = new C();
        d.disC();
        d.disA();}}
```

output

C
A

Hybrid

```
Class A {
    void disA() {
    System.out.println("A");}}
Class B extends A {
     void disB(){
System.out.println("B");}}
Class C extends B{
     void disC(){
System.out.println("C");}}
Class D extends B {
     void disD(){
   System.out.println("D");}}
Class main{
   Public static void main(String[] args){
          D obj = new D();
           d.disA();
           d.disB();
           d.disC();}}
```

Output :-

A
B
C

Multiple

```
Class A{
    int a;
    A(){
       a=5;}
    void display A(){
       System.out.println(a);}}
```

```
interface B {
        int b = 10;
        void displayB();}
Class C extends A implements B {
        int c;
        C(){ c = 15 ;}
        Public void displayB(){
          System.out.println(b);}
        Public void displayC(){
          System.out.println(c);}}
Public Class Main{
        Public Static void main (String[] args){
              C d = new C();
              d.displayA();
              d.displayB();
              d.displayC();}}
```

Output

5
10
15

Exception handling

Class not found exception.

```
Public Class Main {
        Public Static void main (String args[]){
        try { Class.forName("A");}
        catch (ClassNotFoundException e) {
            e.printStackTrace();}}}
```

2. Arithemetic Exception

```
Public Class Main {
    Public Static void main (String [] args ){
        try {
            int a = 10 / 0;
        System.out.println ("code in try Block ");}
        catch (Arithmetic Exception e){
            System.out.println ("Error: Arithmetic error");}}}
```

Output :-

Error: Arithmetic Error.

3. Array Index Out Of Bound

```
Class Main {
    Public Static void main (String [] args ){
        try{
            int a[] = {1, 2, 3};
            System.out.println (a[12]);}
        catch (ArrayIndexOutOfBoundsException e) {
System.out.println ("ArrayIndexOutOfBoundsException : " + e.getMessage());}}
```

Output :-

ArrayIndexOutOfBoundsException : Index 12 out of Bounds.

4) NO Such Method

```
Public Class Main {
    Public Static void main (String args[]){
    try {
        methodThatDoesNotExist();}
    catch (NoSuchMethodException e ) {
        System.out.println ("Caught NoSuchMethodException: "+
        e.getMessage());}}}
```

(5) Illegal Argument Exception.

```
Public Class Person {
    int age;
    Public void setAge (int age) {
        if (age <18) {
            throw new illegalArgumentException ("Age must
be greater than 18 to vote");}
        else {
            this.age = age;
            System.out.println ("The person's age is: " + this.age);
            }}
        Public static void main (String[] args){
            Person p = new Person ();
            try {
                Person.setAge(15);}
            catch (IllegalArgumentException e) {
System.out.println ("Caught an exception: " + e.getMessage());}}}
```

Output:-

Caught an exception: Age must be greater than 18 to vote.