Name – **Kunal Dhanawade**

PRN – **230944520039**

1. Define a class “MyClass”. Define a class “Everything” and define a member function with variable no. of argument inside it. Define a class Demo1 and define main function in it. From main function, call the function of class Everything and make sure you can pass any arguments, it will take and display. It should even take instance of “MyClass” and display.

|  |  |  |  |
| --- | --- | --- | --- |
| |  | | --- | | **public** **class** MyClass {  **private** String name;  **public** MyClass(String name) {  **this**.name = name;  }    **public** String toString() {  **return** "["+name+"]";  }  } | | **public** **class** Everything {  **public** **void** displ(**int** ...n) {  **for**(**int** i=0; i<n.length; i++) {  System.***out***.print(n[i]);  **if** (i!=n.length-1)  System.***out***.print(", ");  }  System.***out***.println();  }  } | | **public** **class** Demo1 {  **public** **static** **void** main(String[] args) {    Everything e = **new** Everything();  e.displ(1);  e.displ(1,0.3);  e.displ(1,0.3,**true**);  e.displ(1,0.3,**true**, "kunal");  e.displ(1,0.3,**true**, "kunal", **new** MyClass("Kunal"));  }  } | |
|  |
| 1  1, 0.3  1, 0.3, true  1, 0.3, true, kunal  1, 0.3, true, kunal, [Kunal] |

1. Define a class Sample with member variable “char ch” and a parameterized constructor. Create some instances (at least 2) of this class by passing any alphabet. Now achieve following tasks:
   1. when you print the reference, it should be displaying the character which was passed during instantiation.
   2. if two instances have same character or an alphabet, their "equals ()" should return true and obviously "hashCode()" also should be same.

|  |
| --- |
| **class** Sample {  **private** **char** ch;  **public** **char** getCh() {  **return** ch;  }  **public** Sample(**char** ch) {  **this**.ch = ch;  }    @Override  **public** String toString() {  // **TODO** Auto-generated method stub  **return** "["+ch+"]";  }    @Override  **public** **boolean** equals(Object obj) {  // **TODO** Auto-generated method stub  Sample ref = (Sample) obj;  **return** **this**.ch == ref.ch;  }    @Override  **public** **int** hashCode() {  // **TODO** Auto-generated method stub  **return** **this**.getCh();  }  }  **public** **class** Demo {  **public** **static** **void** main(String[] args) {  Sample s1 = **new** Sample('v');  System.***out***.println("s1\t"+s1);    Sample s2 = **new** Sample('v');  System.***out***.println("s2\t"+s2);    Sample s3 = **new** Sample('r');  System.***out***.println("s3\t"+s3);    System.***out***.println("s1 == s2\t"+s1.equals(s2));  System.***out***.println("s1 == s3\t"+s1.equals(s3));  System.***out***.println("hashCode of s1\t"+s1.hashCode());  System.***out***.println("hashCode of s2\t"+s2.hashCode());  System.***out***.println("hashCode of s3\t"+s3.hashCode());  }  } |
|  |
| s1 [v]  s2 [v]  s3 [r]  s1 == s2 true  s1 == s3 false  hashCode of s1 118  hashCode of s2 118  hashCode of s3 114 |

1. create 4 classes A, B, C, D with member functions Afun(), Bfun(), Cfun() and Dfun() respectively. now define "Demo" class with main function. Inside main function, create an array of Object class which will store instances of above-mentioned classes. Traverse through this array and call "Cfun ()" of class C only.

|  |
| --- |
| **class** A {  **public** **void** Afun() {  System.***out***.println("in Afun");  }  }  **class** B {  **public** **void** Bfun() {  System.***out***.println("in Bfun");  }  }  **class** C {  **public** **void** Cfun() {  System.***out***.println("in Cfun");  }  }  **class** D {  **public** **void** Dfun() {  System.***out***.println("in Dfun");  }  }  **public** **class** Demo {  **public** **static** **void** main(String[] args) {    Object ob[] = {  **new** A(),  **new** B(),  **new** C(),  **new** D()  };    **for**(Object ref: ob) {  **if**(ref **instanceof** C) {  C c = (C) ref;  c.Cfun();  }  }  }  } |
|  |
| in Cfun |

1. create a parent class "Artist" with a member function "void perform ()". now derive following child classes from "Artist" - Actor, Singer and Musician and override "perform" in these classes to write the specific task.

inside "Actor" class define one more method

void changeGetUp()

{

S.o.p("as per the role demands");

}

now create a Demo class with main function.

inside main function, create array of "Artist" with size 3.

store instances of the above child classes in it.

Now traverse through the array and invoke "perform" of each child class. In addition to this wherever "Actor" object is there inside the array, also perform "changeGetUp()" method.

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| |  | | --- | | **public** **class** Artist {  **public** **void** perform() {  System.***out***.println(**this**.getClass().getName()+" is performing");  }  } | | **public** **class** Actor **extends** Artist {  @Override  **public** **void** perform() {  // **TODO** Auto-generated method stub  **super**.perform();  System.***out***.println("excellent acting");  }    **void** changeGetUp()  {  System.***out***.println("ghanging getup as per the role demands");  }  } | | **public** **class** Singer **extends** Artist {  @Override  **public** **void** perform() {  // **TODO** Auto-generated method stub  **super**.perform();  System.***out***.println("pleasent voice");  }  } | | **public** **class** Musician **extends** Artist {  @Override  **public** **void** perform() {  // **TODO** Auto-generated method stub  **super**.perform();  System.***out***.println("amazing beats");  }  } | | **public** **class** Demo {  **public** **static** **void** main(String[] args) {    Artist art[] = **new** Artist[3];  art[0] = **new** Actor();  art[1] = **new** Singer();  art[2] = **new** Musician();    **for**(**int** i=0; i<art.length; i++) {  art[i].perform();  **if**(art[i] **instanceof** Actor) {  Actor a = (Actor) art[i];  a.changeGetUp();  }  System.***out***.println();  }  }  } | |
|  |
| Actor is performing  excellent acting  ghanging getup as per the role demands  Singer is performing  pleasent voice  Musician is performing  amazing beats |

1. Code this: it's a Developer- Client example

on a developer side:

create a checked exception "VotingNotAllowedException".

define a class "Person" with an instance member "private String name" and "private int age"

define parameterized constructor and toString() method also.

In the parameterized constructor check the age passed while instantiating the class. if the age is less than 18, constructor should raise "VotingNotAllowedException" [it should not handle the exception]

create jar file and documentation

on Client side:

Define a class "Demo" with main function.

From main function create an object of "Person" and display that object.

|  |  |  |
| --- | --- | --- |
| |  | | --- | | /\*\*  \*  \*/  **package** devpack;  /\*\*  \* **@author** kunal  \*  \*/  **public** **class** VotingNotAllowedException **extends** Exception {    **public** VotingNotAllowedException(){  **super**("Under 18 - Voting Not Allowed");  }  } | | /\*\*  \*  \*/  **package** devpack;  /\*\*  \* **@author** kunal  \*  \*/  **public** **class** Person {  **private** String name;  **private** **int** age;    /\*\*  \* **@param** name  \* **@param** age  \* **@throws** VotingNotAllowedException  \*/  **public** Person(String name, **int** age) **throws** VotingNotAllowedException {  **this**.name = name;    **if**(age>18)  **throw** **new** VotingNotAllowedException();  **else**  **this**.age = age;  }    @Override  **public** String toString() {  // **TODO** Auto-generated method stub  **return** "[name: "+name+" age: "+age+"]";  }  } | |
| **package** clientpack;  **import** devpack.Person;  **import** devpack.VotingNotAllowedException;  **public** **class** Demo {  **public** **static** **void** main(String[] args) {  // **TODO** Auto-generated method stub    **try** {  Person p1 = **new** Person("roohi", 17);  System.***out***.println(p1+" addded");  } **catch** (VotingNotAllowedException e) {  // **TODO** Auto-generated catch block  System.***out***.println(e);  }    **try** {  Person p2 = **new** Person("vinay", 24);  System.***out***.println(p2+" addded");  } **catch** (VotingNotAllowedException e) {  // **TODO** Auto-generated catch block  System.***out***.println(e);  }  }  } |
| devpack.VotingNotAllowedException: Under 18 - Voting Not Allowed  [name: vinay age: 24] addded |

1. write a program to display the method names inside the class which is passed as a command line argument.

|  |
| --- |
| **import** java.lang.reflect.Method;  **public** **class** Demo {  **public** **static** **void** main(String[] args) {    Class c=**null**;    **try** {  c=Class.*forName*(args[0]);  } **catch** (ClassNotFoundException e) {  // **TODO** Auto-generated catch block  e.printStackTrace();  }    Method methods[] = c.getDeclaredMethods();    **for**(Method i: methods)  System.***out***.println(i);  }  } |
|  |
| protected void java.lang.Object.finalize() throws java.lang.Throwable  public final void java.lang.Object.wait(long,int) throws java.lang.InterruptedException  public final void java.lang.Object.wait() throws java.lang.InterruptedException  public final native void java.lang.Object.wait(long) throws java.lang.InterruptedException  public boolean java.lang.Object.equals(java.lang.Object)  public java.lang.String java.lang.Object.toString()  public native int java.lang.Object.hashCode()  public final native java.lang.Class java.lang.Object.getClass()  protected native java.lang.Object java.lang.Object.clone() throws java.lang.CloneNotSupportedException  public final native void java.lang.Object.notify()  public final native void java.lang.Object.notifyAll() |

1. on the developer side create 4 classes [ First, Second, Third and Fourth] with only "toString()" method which should return the respective class name.

create necessary jar file and documentation

on the client side

create a class "Demo" with main function and

static Object returnObject(String name) method.

this "returnObject()" method should create the object of the given inside "name" parameter using Reflection API and return it.

from the main function invoke "returnObject()" by passing name of either "First" or "Second" or "Third" or "Fourth" only. Then main function should collect the object given by "returnObject()" method and display it.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| |  | | --- | | /\*\*  \*  \*/  **package** devpack;  /\*\*  \* **@author** kunal  \*  \*/  **public** **class** First {  @Override  **public** String toString() {  // **TODO** Auto-generated method stub  **return** "[ "+**this**.getClass().getName()+" ]";  }  } | | /\*\*  \*  \*/  **package** devpack;  /\*\*  \* **@author** kunal  \*  \*/  **public** **class** Second {  @Override  **public** String toString() {  // **TODO** Auto-generated method stub  **return** "[ "+**this**.getClass().getName()+" ]";  }  } | | /\*\*  \*  \*/  **package** devpack;  /\*\*  \* **@author** kunal  \*  \*/  **public** **class** Third {  @Override  **public** String toString() {  // **TODO** Auto-generated method stub  **return** "[ "+**this**.getClass().getName()+" ]";  }  } | | /\*\*  \*  \*/  **package** devpack;  /\*\*  \*  \*/  **public** **class** Fourth {  @Override  **public** String toString() {  // **TODO** Auto-generated method stub  **return** "[ "+**this**.getClass().getName()+" ]";  }  } | |
| **package** clientpack;  **public** **class** Demo {  **static** Object returnObject(String name) {    Object ob = **null**;    **try** {  Class classDefinition = Class.*forName*(name);  ob = classDefinition.~~newInstance~~();  } **catch** (InstantiationException e) {  e.printStackTrace();  } **catch** (IllegalAccessException e) {  e.printStackTrace();  } **catch** (ClassNotFoundException e) {  e.printStackTrace();  }    **return** ob;  }    **public** **static** **void** main(String[] args) {    String objects[] = {  "devpack.First",  "devpack.Second",  "devpack.Third",  "devpack.Fourth"  };    **for**(String s: objects) {  Object ob = *returnObject*(s);  System.***out***.println(ob);  }  }  } |
| [ devpack.First ]  [ devpack.Second ]  [ devpack.Third ]  [ devpack.Fourth ] |