1. 20 Balık adında bir sınıf hazırlayıp UML diyagramını hazırlayınız. Balığın Büyüklüğü vardır [1-10], balığın avlarını yakalama olasılıkları vardır [0.30-0.70]. Balık canlı olup olmadığını bize söyler. BalıkAvlanmasi adından bir arabirim kullanır ve bu arabirim içinde void Avlan(Balik b) metodu bulunmaktadır. Bu balıktan türeyen Köpek balığı ve Japon Balığı türetiniz UML diyagramına ekleyiniz.
2. 40 Balıklar doğduklarında en küçük boydadır. Bir balığın diğer balığı yemesi için boyunun avlandığı diğer balıktan büyük yada aynı olması ve yakalama olasılığının avlanmanın balığın yenmesi ile sonuçlanıp sonuçlanmayacağı anlamına gelmektedir. Eğer şansı tutarsa balığı yiyebilir. Bir balık yiyen balığın boyu bir artar. Yenilen balık ölür. Ölü balık tekrardan yenilemez.
Bu bilgiler ışığında bir havuz içinde 10 tane yeni balık doğmaktadır, bu balıklar birbirleri ile tek balık kalıncaya kadar birbirlerini yemeye devam ederler. Buna göre balıkların yenmesi ile ilgili simülasyon programını hazırlayınız.
3. 10 Aşağıdaki Program Çalıştırıldığında Ekrana ne çıkarır.

class EnclosingClass {

 private int x=0,y=0;

 public EnclosingClass() {

 }

 class MemberClass {

 private int x = 4;

 private int y= 4;

 public void aMethod( int x ) {

 System.out.println(x);

 // 10 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 System.out.println(this.x);

 // 4\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 System.out.println(EnclosingClass.this.x);

 //0\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 System.out.println(y);

 // 4\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

 }

 }

 public void aMethod() {

 MemberClass mem = new MemberClass ();

 mem.aMethod( 10 );

 }

}

public class TestClassMember {

 public static void main(String[ ] args) {

 EnclosingClass e = new EnclosingClass();

 e.aMethod();

 }

}

1. 10 Aşağıdaki kodlar çalıştırıldığında ekrana ne yazacaktır. Neden ?

class Xplain {

 void print() { System.out.println(“A”); }

 }

 class Xtend extends Xplain {

 void print() { System.out.println(“B”); }

 }

 public class Question2 {

 public static main(String args[ ]) {

 Xplain xp1 = new Xplain();

 Xtend xt2 = new Xtend();

 Xtend xt3;

 xp1.print(); //A \_\_\_\_\_\_Çünkü Xplain sınıfına referans verir\_\_\_\_\_\_\_\_\_\_\_\_\_

 xt2.print(); // B\_\_\_\_\_\_ Çünkü Xtend nesnensine referans verir\_\_\_\_\_\_\_\_\_\_\_\_\_

 xp1 = xt2;

 xp1.print(); // B\_\_\_\_\_\_\_ Çünkü Xtend nesnesine REf\_\_\_\_\_\_\_\_\_\_\_\_

 xt3 = (Xtend) xp1;

 xt3.print(); // B\_\_\_\_\_\_\_ Çünkü Xtend nesnesine REf \_\_\_\_\_\_\_\_\_\_\_\_

 ((Xtend) xp1).print(); // B\_\_\_\_\_\_\_ Çünkü Xtend nesnesine REf \_\_\_\_\_\_\_\_\_\_\_\_

 }

 }

1. 10 Aşağıdaki Kod derlenmemektedir. Düzeltip Çalıştırınız.

public class OuterClass {

 private static int aStaticVariable;

 private int anInstanceVariable;

 public void anInstanceMethod() {

 System.out.println(“anInstanceMethod”);

 }

 public static void aStaticMethod() {

 System.out.println(“aStaticMethod”);

 }

 public static class StaticMember {

 public void aMethod() {

 int staticValue = OuterClass.aStaticVariable;

 OuterClass.aStaticMethod();

 int value = anInstanceVariable;

 OuterClass.anInstanceMethod();

 }

 }

 public class Member {

 public void aMethod() {

 int staticValue = aStaticVariable;

 int value = anInstanceVariable;

 aStaticMethod();

 anInstanceMethod();

 }

 }

}

iki Sınıfın UML diyagramları verilmiştir. Buna göre shape adında soyut bir sınıf oluşturulmak istenmektedir. Bu Sınıflardan türeyecek olan Circle ve REctangle için UML i tekrardan çiziniz.

|  |
| --- |
| Circle |
| * color : int
* hPosition : int
* vPosition : int
* radius : double
 |
| * + draw()
	+ resize()
	+ setColor()
	+ move()
 |

|  |
| --- |
| Rectangle |
| * color : int
* hPosition : int
* vPosition : int
* width : double
* height : double
 |
| * + draw()
	+ resize()
	+ setColor()
	+ move()
 |

1. 10