22. Prikazati stanje memorije nakon što se izvršila linija sa oznakom 1. Koliko će objekata *garbage collector* počistiti?

// Tester.java

**class** Beta {

}

**class** Alpha {

**static** Beta b1;

Beta b2;

}

**public** **class** Tester {

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

Beta b1 = **new** Beta();

Beta b2 = **new** Beta();

Alpha a1 = **new** Alpha();

Alpha a2 = **new** Alpha();

a1.b1 = b1;

a1.b2 = b1;

a2.b2 = b2;

a1 = null;

b1 = null;

b2 = null; // 1

}

}

26. Prikazati stanje memorije u trenutku izvršavanja linije sa oznakom 1. Pretpostaviti da je rezervisana dovoljna veličina *heap*-a i da se *garbage collector* odmah izvršava nakon poziva metode *gc()*.

// Memory.java

**public** **class** Memory {

**int** id;

**double**[] doubles = **new** double[10000000];

**long**[] longs = **new** long[6000000];

Memory[] mys = **new** Memory[3];

Memory m;

Dumb d;

**public** Memory(Memory m) {

**this**.m = m;

**if**(m!=**null**)

id = m.id + 1;

}

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

Memory m0 = **new** Memory(**null**);

Memory m1 = **new** Memory(m0);

Memory m2 = **new** Memory(m1);

Memory m3 = **new** Memory(m2);

Memory m4 = m3;

Memory m5 = **new** Memory(m0);

Memory[] ms = **new** Memory[3];

ms[0] = **new** Memory(m2);

ms[1] = **new** Memory(m3);

ms[2] = **new** Memory(m2);

m1 = m2 = m3 = m5 = null;

System.gc();

ms[0] = **new** Memory(ms[0]);

ms[1] = **new** Memory(ms[1]);

ms[2] = **new** Memory(ms[2]);

ms[2].m = ms[0];

m3 = **new** Memory(ms[0] = **null**);

ms[0] = **new** Memory(ms[0]);

System.gc(); // 1

}

}

**class** Dumb {

**float**[] floats = **new** float[20000000];

**int**[] ints = **new** int[40000000];

}

27. Napisati izlaz sljedećeg programa i prikazati stanje memorije u trenutku izvršavanja linije sa oznakom 1. Pretpostaviti da je rezervisana dovoljna veličina *heap*-a i da se *garbage collector* odmah izvršava nakon poziva metode *gc()*.

**Napomena:** metoda *finalize()* se poziva prilikom čišćenja objekta sa *heap*-aod strane *garbage collector*-a.

// M1.java HEAP=500MB

**public** **class** M1 {

**int** id;

M1 m1;

M2 m2;

**char**[] nizChar = **new** char[10\_000\_000];

**int**[] nizInt = **new** int[5\_000\_000];

M2 mArray[][] = **new** M2[3][2];

**public** M1(int id) {

System**.out.**println("M1: " + id);

**this**.id = id;

}

**public** M1(M1 m1,**int** id,M2 m2) {

System**.out.**println("M1: " + id);

**this**.m1 = m1;

**this**.m2 = m2;

**this**.id = id;

}

@Override

**protected** **void** finalize() {

System**.out.**println(id + "finallize");

}

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

M1 m10 = **new** M1(10);

M2 m21 = **new** M2(m10,21);

M1 m11 = **new** M1(m10,11,m21);

M2 m22 = **new** M2(**null**,22);

M1 m12 = **new** M1(**null**,12,m22);

m12.mArray[0][0] = m22;

m12.mArray[1][1] = **new** M2(23);

m12.mArray[2][1] = **new** M2(24);

m12.mArray[1] = **null**;

System.gc();

m10.mArray[1][0] = **new** M2(25);

m11.m2 = m10.mArray[1][0];

m11 = **null**;

System.gc();

m10.mArray[2][2] = **new** M2(**new** M1(1000),26);

System.gc(); // 1

}

}

**class** M2 {

**float**[] f = **new** float[2\_500\_000];

M1 m1 = **new** M1(0);

**private** **int** id2 = 0;

**public** M2(M1 m1,**int** id2) {

System**.out.**println("M2: " + id2);

**this**.m1 = m1;

**this**.id2 = id2;

}

**public** M2(**int** id2) {

System**.out.**println("M2: " + id2);

**this**.id2 = id2;

}

@Override

**protected** **void** finalize() {

System**.out.**println(id2 + "finallize");

}

}

00.

public class Mainy{

int id;

double[] doubles = new double[10000000];

long[] longs = new long[6000000];

Mainy[] mys = new Mainy[3];

Mainy m;

Dumb d;

public Mainy(Main m){

this.m = m;

if(m!=null)

id = m.id + 1;

}

public static void main(String[] args){

Mainy m0 = new Mainy(null);

Mainy m1 = new Mainy(m0);

Mainy m2 = new Mainy(m1);

Mainy m3 = new Mainy(m2);

Mainy m4 = m3;

Mainy m5 = new Mainy(m0);

Mainy[] ms = new Mainy[3];

ms[0] = new Mainy(m2);

ms[1] = new Mainy(m3);

ms[2] = new Mainy(m2);

m1 = m2 = m3 = m5 = null;

System.gc(); //1 //pozivanje garbage collectora

ms[0] = new Mainy(ms[0]);

ms[1] = new Mainy(ms[1]);

ms[2] = new Mainy(ms[2]);

ms[2].m = ms[0];

m3 = new Mainy(ms[0] = null);

ms[0] = new Mainy(ms[0]);

System.gc(); //2

}

}

class Dumb {

float[] flaots = new float[20000000];

int[] ints = new int[40000000];

}

01.

2000 MB heap

public class Mainy{

public static k = true;

int id;

double[] doubles = new double[12\_000\_000];

long[] longs = new long[8\_000\_000];

Dumb[][] matrica = new Dumb[3][2];

Dumb d;

Mainy m;

public Mainy(){};

public Mainy(int id){

if(m!=null)

id = m.id + 1;

}

public Mainy(Dumb d,Mainy m,int id){

this.m = m;

this.d = d;

this.id = id;

if(k =! k)

new Mainy()

else{

Dumb db = new Dumb(null,0);

}

}

public static void main(String[] args){

Mainy m0 = new Mainy(10);

Dumb d0 = new Dumb(m0,11);

Mainy m2 = new Mainy(d0,m0,12);

m0.matrica[0][0] = new Dumb(m2,12);

m0.matrica[0][1] = new Dumb(null,13);

m2.d = m0.matrica[0][0];

d0.m = null;

m0.matrica[0][0] = null;

m0.matrica[1][0] = new Dumb(m0,11);

m0.matrica[2][0] = d0;

m0.matrica[2][1] = m0.matrica[0][0];

m2.matrica[0][1] = new Dumb(new Mainy(15));

====================================================

Mainy m3 = new Mainy(m2);

Mainy m4 = m3;

Mainy m5 = new Mainy(m0);

Mainy[] ms = new Mainy[3];

ms[0] = new Mainy(m2);

ms[1] = new Mainy(m3);

ms[2] = new Mainy(m2);

m1 = m2 = m3 = m5 = null;

System.gc(); //1 //pozivanje garbage collectora

ms[0] = new Mainy(ms[0]);

ms[1] = new Mainy(ms[1]);

ms[2] = new Mainy(ms[2]);

ms[2].m = ms[0];

m3 = new Mainy(ms[0] = null);

ms[0] = new Mainy(ms[0]);

System.gc(); //2

}

}

class Dumb {

float[] flaots = new float[20\_000\_000];

int[] ints = new int[40\_000\_000];

int id = 0;

Mainy m = new Mainy(999);

public Dumb(Mainy m,int id){

this.m = m;

this.id = id;

}

}