**Part 1**

import java.util.Arrays;

import java.util.concurrent.ThreadLocalRandom;

public class Part1 {

public static void main(String[] args) {

// TODO Auto-generated method stub

int[] stats = {0,0,0,0,0,0,0};

int roll = 0;

for(int i = 0; i < 100; i++) {

roll = ThreadLocalRandom.current().nextInt(1, 7);

switch(roll) {

case 1:

stats[1] = stats[1] + 1;

break;

case 2:

stats[2] = stats[2] + 1;

break;

case 3:

stats[3] = stats[3] + 1;

break;

case 4:

stats[4] = stats[4] + 1;

break;

case 5:

stats[5] = stats[5] + 1;

break;

case 6:

stats[6] = stats[6] + 1;

break;

}

}

System.out.println(Arrays.toString(stats));

}

}

**Output**

[0, 11, 22, 16, 9, 26, 16]

**Part 2**

import java.util.Arrays;

import java.util.concurrent.ThreadLocalRandom;

public class Part1 {

public static void main(String[] args) {

// TODO Auto-generated method stub

int[] stats = {0,0,0,0,0,0,0};

int roll = 0;

for(int i = 0; i < 100; i++) {

roll = ThreadLocalRandom.current().nextInt(1, 7);

stats[roll] = stats[roll] + 1;

}

for(int i = 1; i < 7; i++) {

System.out.print("[" + i + "]\t");

for(int j = 0; j < stats[i]; j++) {

System.out.print("\*");

}

System.out.print(" " + stats[i]);

System.out.println("");

}

}

}

**Output**

[1] \*\*\*\*\*\*\*\*\*\*\*\* 12

[2] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 24

[3] \*\*\*\*\*\*\*\* 8

[4] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 19

[5] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 22

[6] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 15

**Part 3**

import java.util.Arrays;

import java.util.concurrent.ThreadLocalRandom;

public class Part1 {

public static void main(String[] args) {

// TODO Auto-generated method stub

int[] stats = new int[13];

int roll1, roll2, sum;

for(int i = 0; i < 100; i++) {

roll1 = ThreadLocalRandom.current().nextInt(1, 7);

roll2 = ThreadLocalRandom.current().nextInt(1, 7);

sum = roll1 + roll2;

stats[sum] = stats[sum] + 1;

}

for(int i = 2; i < stats.length; i++) {

System.out.print("[" + i + "]\t");

for(int j = 0; j < stats[i]; j++) {

System.out.print("\*");

}

System.out.print(" " + stats[i]);

System.out.println("");

}

}

**Output**

[2] \*\* 2

[3] \*\*\*\* 4

[4] \*\*\*\*\*\*\*\*\*\*\* 11

[5] \*\*\*\*\*\*\*\*\*\*\* 11

[6] \*\*\*\*\*\*\*\*\*\* 10

[7] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 22

[8] \*\*\*\*\*\*\*\*\*\*\*\*\* 13

[9] \*\*\*\*\*\*\*\*\*\*\*\*\*\* 14

[10] \*\*\*\*\*\*\*\* 8

[11] \*\*\*\* 4

[12] \* 1

**Part 4**

import java.util.Arrays;

import java.util.concurrent.ThreadLocalRandom;

public class Part1 {

public static void main(String[] args) {

// TODO Auto-generated method stub

int[] stats = new int[13];

int roll1, roll2, sum, rolls = 30000, scale = 100;

for(int i = 0; i < rolls; i++) {

roll1 = ThreadLocalRandom.current().nextInt(1, 7);

roll2 = ThreadLocalRandom.current().nextInt(1, 7);

sum = roll1 + roll2;

stats[sum] = stats[sum] + 1;

}

for(int i = 2; i < stats.length; i++) {

System.out.print("[" + i + "]\t");

for(int j = 0; j < (stats[i] +(100 - (stats[i] % scale))) / scale; j++) {

System.out.print("\*");

}

System.out.print(" " + stats[i]);

System.out.println("");

}

}

}

**Output**

[2] \*\*\*\*\*\*\*\*\* 829

[3] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1695

[4] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2499

[5] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3356

[6] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 4111

[7] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 5056

[8] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 4144

[9] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3346

[10] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2439

[11] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1675

[12] \*\*\*\*\*\*\*\*\* 850

**Part 5**

import java.util.concurrent.ThreadLocalRandom;

public class Part1 {

public static void main(String[] args) {

// TODO Auto-generated method stub

int[] stats = new int[19];

int roll1, roll2, roll3, sum, rolls = 30000, scale = 100;

for(int i = 0; i < rolls; i++) {

roll1 = ThreadLocalRandom.current().nextInt(1, 7);

roll2 = ThreadLocalRandom.current().nextInt(1, 7);

roll3 = ThreadLocalRandom.current().nextInt(1, 7);

sum = roll1 + roll2 + roll3;

stats[sum] = stats[sum] + 1;

}

for(int i = 3; i < stats.length; i++) {

System.out.print("[" + i + "]\t");

for(int j = 0; j < (stats[i] +(100 - (stats[i] % scale))) / scale; j++) {

System.out.print("\*");

}

System.out.print(" " + stats[i]);

System.out.println("");

}

}

}

**Output**

[3] \*\* 135

[4] \*\*\*\*\* 473

[5] \*\*\*\*\*\*\*\*\* 859

[6] \*\*\*\*\*\*\*\*\*\*\*\*\*\* 1358

[7] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1955

[8] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2898

[9] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3563

[10] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3676

[11] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3765

[12] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 3545

[13] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 2944

[14] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1999

[15] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 1429

[16] \*\*\*\*\*\*\*\*\* 831

[17] \*\*\*\*\* 425

[18] \*\* 145

**Extra Credit 1 & 2**

import java.util.Scanner;

import java.util.concurrent.ThreadLocalRandom;

public class Part1 {

public static void main(String[] args) {

int rolls, dice, scale;

//ask user how many dice and rolls

Scanner scnr = new Scanner(System.in);

System.out.print("How many dice should be rolled? : ");

dice = scnr.nextInt();

System.out.print("How many times should the dice be rolled? : ");

rolls = scnr.nextInt();

//load rolls into array

int[] stats = roll(rolls, dice);

//find largest value in array

int largeValue = 0;

for(int i = 0; i < stats.length; i++) {

if(stats[i] > largeValue)

largeValue = stats[i];

}

//set scale so that largest value has 50 '\*' in histogram

if(largeValue > 50)

scale = largeValue / 50;

else

scale = 1;

//build histogram

for(int i = dice; i < stats.length; i++) {

System.out.print("[" + i + "]\t");

for(int j = 0; j < stats[i] / scale; j++) {

System.out.print("\*");

}

System.out.print(" " + stats[i]);

System.out.println("");

}

}

//generate rolls

public static int[] roll(int rolls, int dice) {

int[] stats = new int[6\*dice + 1];

for(int i = 0; i < rolls; i++) {

int sum = 0;

for(int j = 0; j < dice; j++) {

sum += ThreadLocalRandom.current().nextInt(1, 7);

}

stats[sum] = stats[sum] + 1;

}

return( stats );

}

}

**Output 1**

How many dice should be rolled? : 1

How many times should the dice be rolled? : 1000000

[1] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 167131

[2] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 167097

[3] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 165842

[4] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 166535

[5] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 166693

[6] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 166702

Scale: 3342

**Output 2**

How many dice should be rolled? : 5

How many times should the dice be rolled? : 1000000

[5] 130

[6] 640

[7] 1905

[8] \*\* 4595

[9] \*\*\*\* 8803

[10] \*\*\*\*\*\*\*\* 16301

[11] \*\*\*\*\*\*\*\*\*\*\*\*\* 26473

[12] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 39593

[13] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 53822

[14] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 69450

[15] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 83682

[16] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 94364

[17] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100309

[18] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 100475

[19] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 94551

[20] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 83991

[21] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 69293

[22] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 53898

[23] \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\* 39084

[24] \*\*\*\*\*\*\*\*\*\*\*\* 26036

[25] \*\*\*\*\*\*\*\* 16341

[26] \*\*\*\* 9026

[27] \*\* 4532

[28] 1913

[29] 661

[30] 132

Scale: 2009