Divisors

```
public class Divisors
{
        public static void main (String[] args)
         {
                int number = Integer.parseInt(args[0]);
                int devisor = 1 ; //set the devisor on one and check all the
devisers that could be
                while (devisor<number + 1)
                {
                        if(number % devisor == 0) //check if the number is devisor
and print him
                                System.out.println(devisor);
                        devisor ++;
                }
        }
}
```

Reverse

```
public class Reverse

{
        public static void main (String[] args)
        {
                String revstring = "";
                String strorigin = args[0];
                for(int i = strorigin.length() - 1; i > -1; i --)
                {
                        revstring = revstring + strorigin.charAt(i);
                }

                System.out.println(revstring);
                System.out.println("The middle character is " +
strorigin.charAt((strorigin.length()-1)/2));



        }
}
```

InOrder

```
public class InOrder

{
    public static void main (String[] args)
    {
        int previousnumber = (int)(Math.random()*10); //genrate number
from 0 to 10
        System.out.println(previousnumber); //print the first number
        boolean inorder = true;
        while (inorder)
        {
            int number = (int)(Math.random()*10);
            if (previousnumber<number) //check if the number that
we genrate is grater than the previous number and print him
            {
                System.out.println(number);
                previousnumber=number; // set the number to be
the previous number
            }
            else inorder=false;
        }

    }
}
```

Reverse

```
public class Reverse
 {
        public static void main (String[] args)
        {
                String revstring = "";//define a new string to be the reverse
                String strorigin = args[0];
                for(int i = strorigin.length() - 1; i > -1; i --)//getting from the last
char to the first one
                {
                        revstring = revstring + strorigin.charAt(i);//run from the
last char of the origin string and add it to new string
                }

                System.out.println(revstring);
                System.out.println("The middle character is " +
strorigin.charAt((strorigin.length()-1)/2));//printing the middle char


        }
}
```

<u>**Perfect**</u>

```
public class Perfect
{
    public static void main (String[] args)
    {
        //// creating sum that i could test if is equal to the number we get
and create deviser that follow all the options of the possible deviser
        int sum = 0 , thedeviser = 1;
        int theperfctumber = Integer.parseInt(args[0]);
        String sequence = ""; //creating sequnce that save all the deviser
in string
        while (thedeviser<theperfctumber)
        {
            if(theperfctumber%thedeviser == 0) //checking if the
cuurent number is deviser
            {
                sum += thedeviser;
                sequence += thedeviser + " + ";
            }
            thedeviser++;
        }

        if(sum == theperfctumber) //checikng if the number is perfect
        System.out.println( theperfctumber+" is a perfect number since "
+ theperfctumber + " = " + sequence.substring(0,sequence.length()-3) );
//printing the sequense without the last + and space
        else System.out.println(theperfctumber + " is not a perfect
number");
    }
}
```

DamkaBoard

```
public class DamkaBoard

{
        public static void main(String[] args)

        {
                int damkanumber= Integer.parseInt(args[0]);
                for (int i = 0;i<damkanumber;i++)
                {
                        for (int j = 0 ; j < damkanumber; j ++) //using for in
for to print both line and queue
                        {
                                if(i % 2 == 0) //check if he is evan line to
order the *
                                System.out.print("* ");
                        else System.out.print(" *");
                        }

                                System.out.println("");
                }
        }
}
```

OneOfEach

```
public class OneOfEach

{
        public static void main (String[] args)
        {
                int count = 0;
                boolean boy = true, girl = true; //checking if i have both girl and
boy
                while (boy || girl)
                {

                        double isgender = Math.random();
                        if(isgender > 0.5)
                        {
                                System.out.print(" g ");
                                girl = false;
                        }

                        else
                        {
                                System.out.print(" b ");
                                boy = false;
                        }
                        count ++;
                }
                System.out.println("");
                System.out.print("You made it... and you now have " + count + "
children.") ;
        }
}
```

OneOfEachStats1

```
public class OneOfEachStats1

 {
        public static void main (String[] args)
        {
                int numberoffamily = Integer.parseInt(args[0]);
                int countotalchildern = 0 , count2ch = 0 , count3ch = 0 ,
count4ch = 0 ;
                for (int i=0;i<numberoffamily;i++) //using my exsist code to
genrate number of childern for family that i got
                {
                int countlocalchild = 0;
                boolean boy = true, girl = true; //checking if i have both girl and
boy
                        while (boy || girl)
                        {

                                double isgender = Math.random();
                                if(isgender > 0.5)
                                {
                                        girl = false;
                                }

                                else
                                {
                                        boy = false;
                                }
                                countlocalchild ++;
                                countotalchildern ++;

                        }

                        if (countlocalchild == 2 ) //checking for which group of
family he belong
                                        count2ch++;
                        if (countlocalchild == 3 )
                                        count3ch++;
                        if (countlocalchild > 3 )
                                        count4ch++;
                }
                double numberoffamilydouble = 0.0 + numberoffamily;
                System.out.println("Average: "+
countotalchildern/numberoffamilydouble +" children to get at least one of each
gender.");
```

```
        System.out.println("Number of families with 2 children:
"+count2ch);
        System.out.println("Number of families with 3 children:
"+count3ch);
        System.out.println("Number of families with 4 or more children:
"+count4ch);
        int commonfamily = Math.max(
(Math.max(count2ch,count3ch)),count4ch);
        if (commonfamily == count2ch)
            System.out.println("The most common number of children
is 2");
            else if (commonfamily == count3ch)
                    System.out.println("The most common
number of children is 3");
                        else if (commonfamily == count4ch)
                        System.out.println("The most
common number of children is 4 or more");


    }
}
```

OneOfEachStats

```
public class OneOfEachStats {
    public static void main (String[] args) {
        // Gets the two command-line arguments
        int T = Integer.parseInt(args[0]);
        int seed = Integer.parseInt(args[1]);
        // Initailizes a random numbers generator with the given seed
value
   Random generator = new Random(seed);
        int numberoffamily = Integer.parseInt(args[0]);
        int countotalchildern = 0 , count2ch = 0 , count3ch = 0 ,
count4ch = 0 ;
        for (int i=0;i<numberoffamily;i++) //using my exsist code to
genrate number of childern for family that i got
        {
        int countlocalchild = 0;
        boolean boy = true, girl = true; //checking if i have both girl and
boy
            while (boy || girl)
            {

                double isgender = generator.nextDouble();
                if(isgender > 0.5)
                {
                    girl = false;
                }

                else
                {
                    boy = false;
                }
                countlocalchild ++;
                countotalchildern ++;

            }

            if (countlocalchild == 2 ) //checking for which group of
family he belong
                    count2ch++;
            if (countlocalchild == 3 )
                    count3ch++;
            if (countlocalchild > 3 )
                    count4ch++;
        }
        double numberoffamilydouble = 0.0 + numberoffamily; //casting
the number of family to double and printing the average number and the
number of each family
        System.out.println("Average: "+
countotalchildern/numberoffamilydouble +" children to get at least one of each
gender.");
        System.out.println("Number of families with 2 children:
"+count2ch);
        System.out.println("Number of families with 3 children:
"+count3ch);
        System.out.println("Number of families with 4 or more children:
"+count4ch);
        int commonfamily = Math.max(
(Math.max(count2ch,count3ch)),count4ch); //checking which of the number of
child is most common
        if (commonfamily == count2ch)
            System.out.println("The most common number of children
is 2.");
            else if (commonfamily == count3ch)
                    System.out.println("The most common
number of children is 3.");
                        else if (commonfamily == count4ch)
                        System.out.println("The most
common number of children is 4 or more.");

    }
}
```