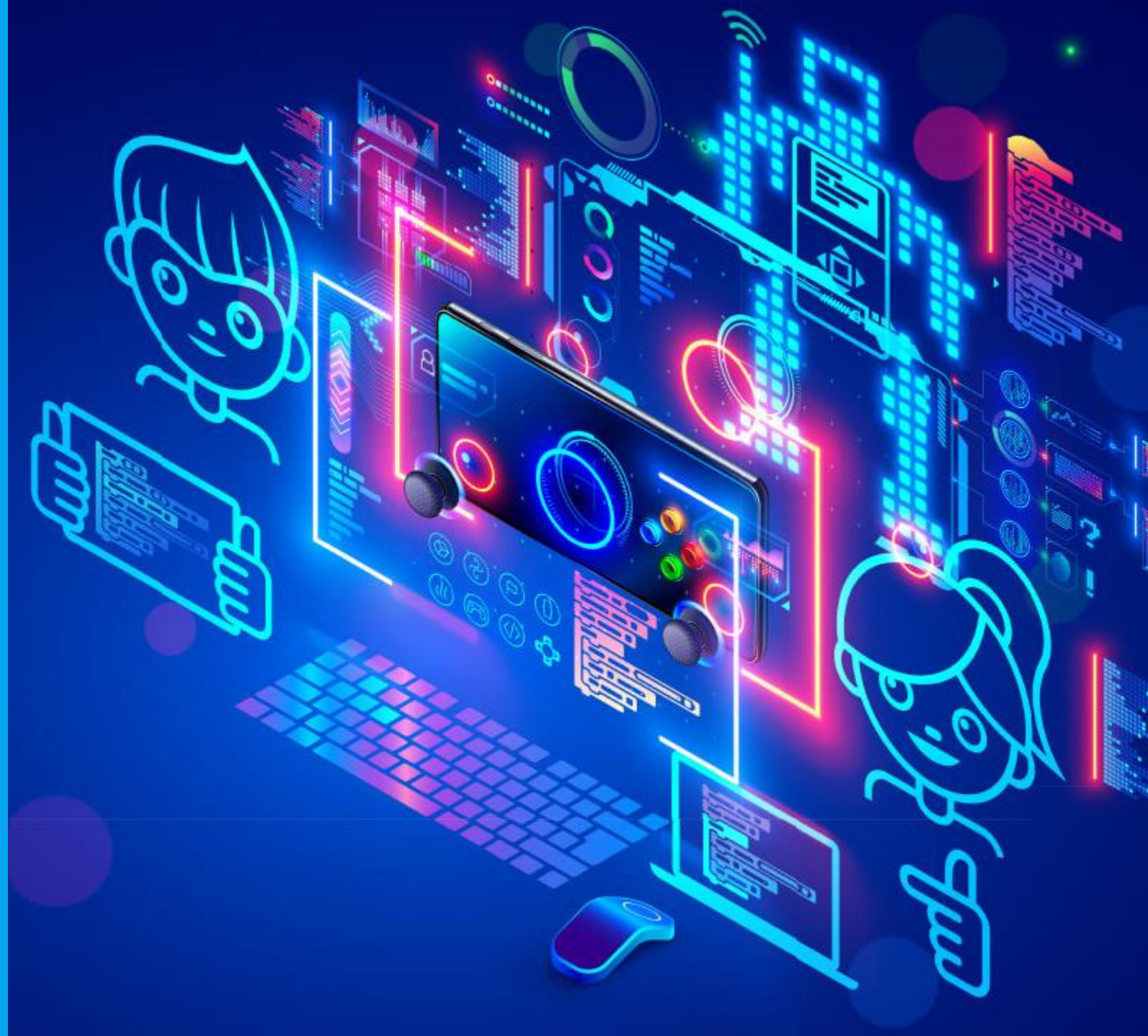
CS 24 AP Computer Science A Review

Week 9: FRQ and Algorithms I

DR. ERIC CHOU
IEEE SENIOR MEMBER

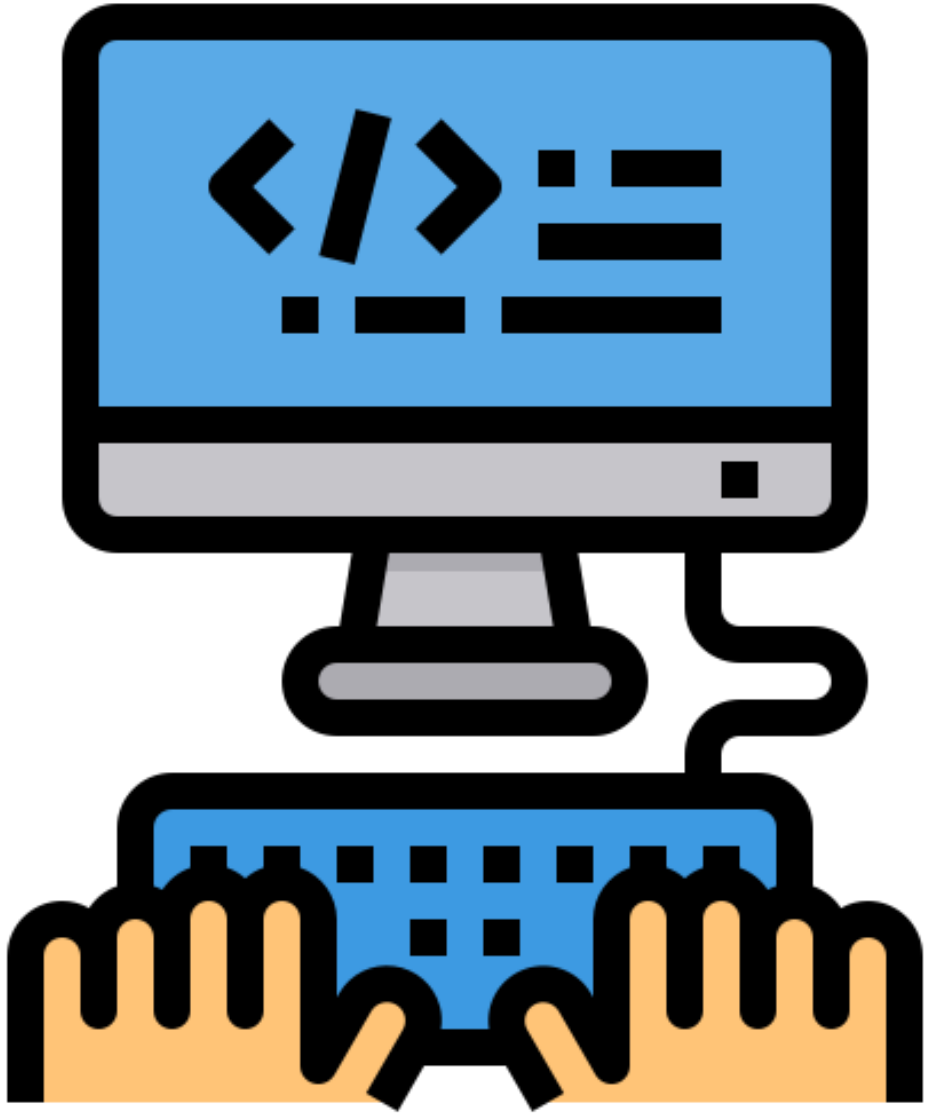


Topics

- Least Integer Function (Ceiling Function), Greatest Integer Function (Floor Function), Rounding Function
- Two-way Toggler, 3-way Toggler
- Use of Modulus Arithmetic, Number of Fridays in a Year.
- Modulus Wrap-around
- Random Number Generation, Choice of Random Element, Biased Random Condition
- Swap, Rotation and Shuffling

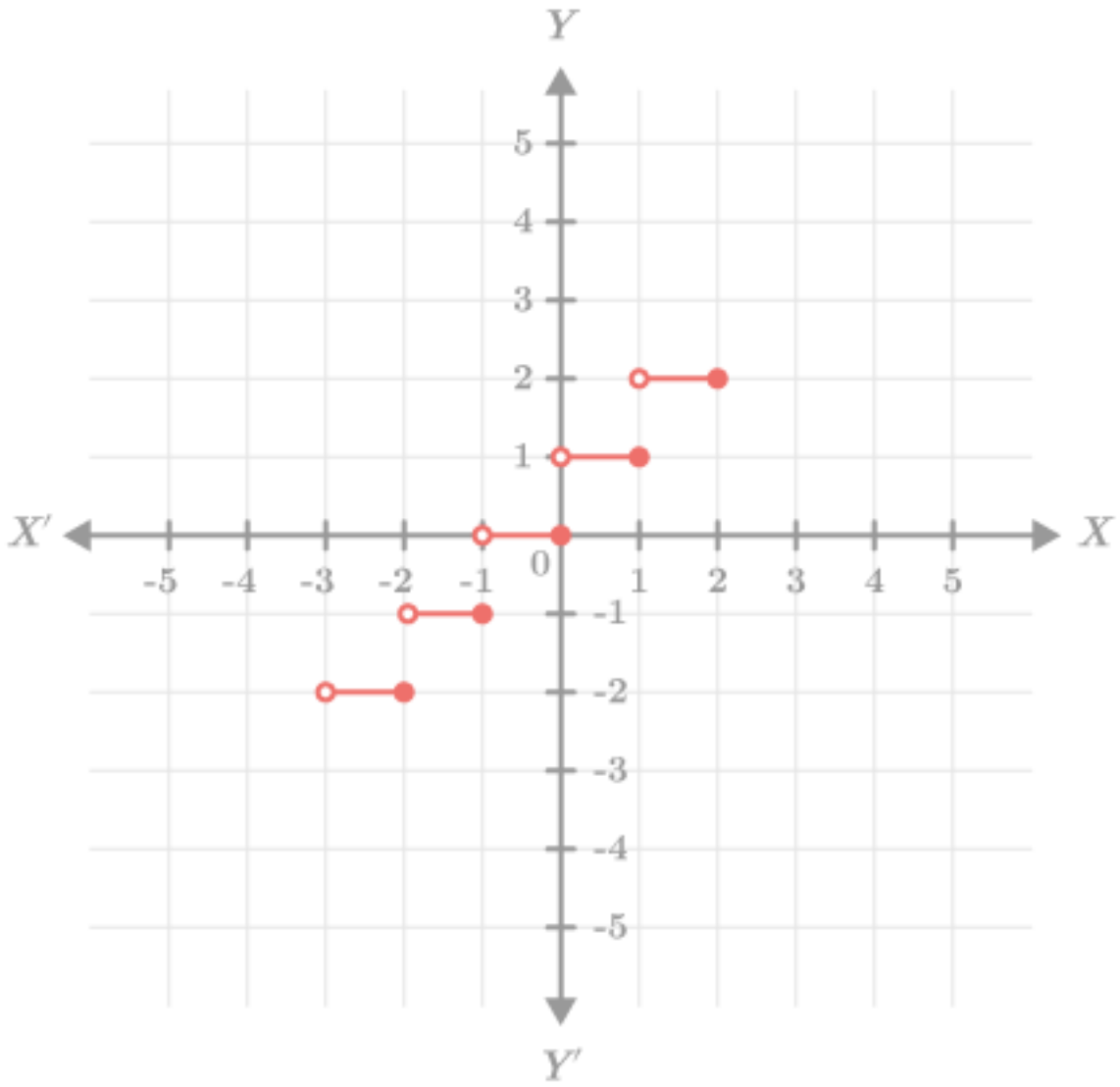
Topics

- Reversal of String and Palindrome, Successive Division (Subtraction)
- Sum of Digits
- Indefinite Loop (Continue-Condition and Exit Condition)
- Windowed Traversal (block detection)
- Traversal By Section (a section of 3)
- Maximum/Minimum, Top 3
- isStrictly increasing, difference array (Discrete Integral, Difference)



Integer Functions

Section 1



Least Integer Function (Ceiling Function)

Wrapping Box Problem

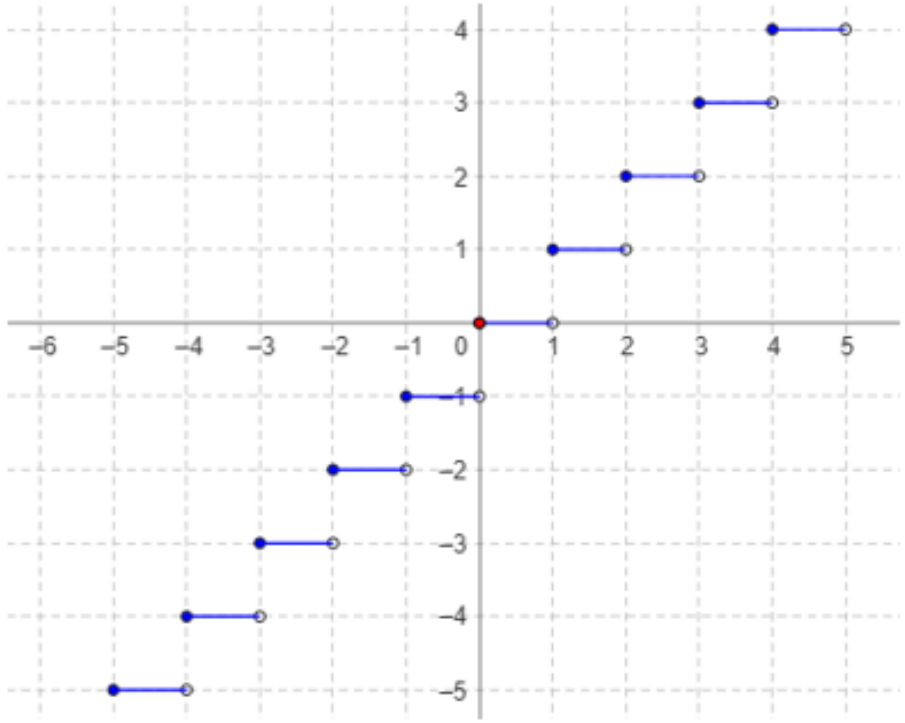
- There are totally 35 cup cakes to be put into boxes of 8 each. How many boxes are needed to store all cup cakes?

```
int x = 35 / 8;
if (35%8 !=0) x++;
```

The Greatest Integer Function

The greatest integer function will round any number down to the nearest integer.

Example: $f(x) = \lfloor x \rfloor$

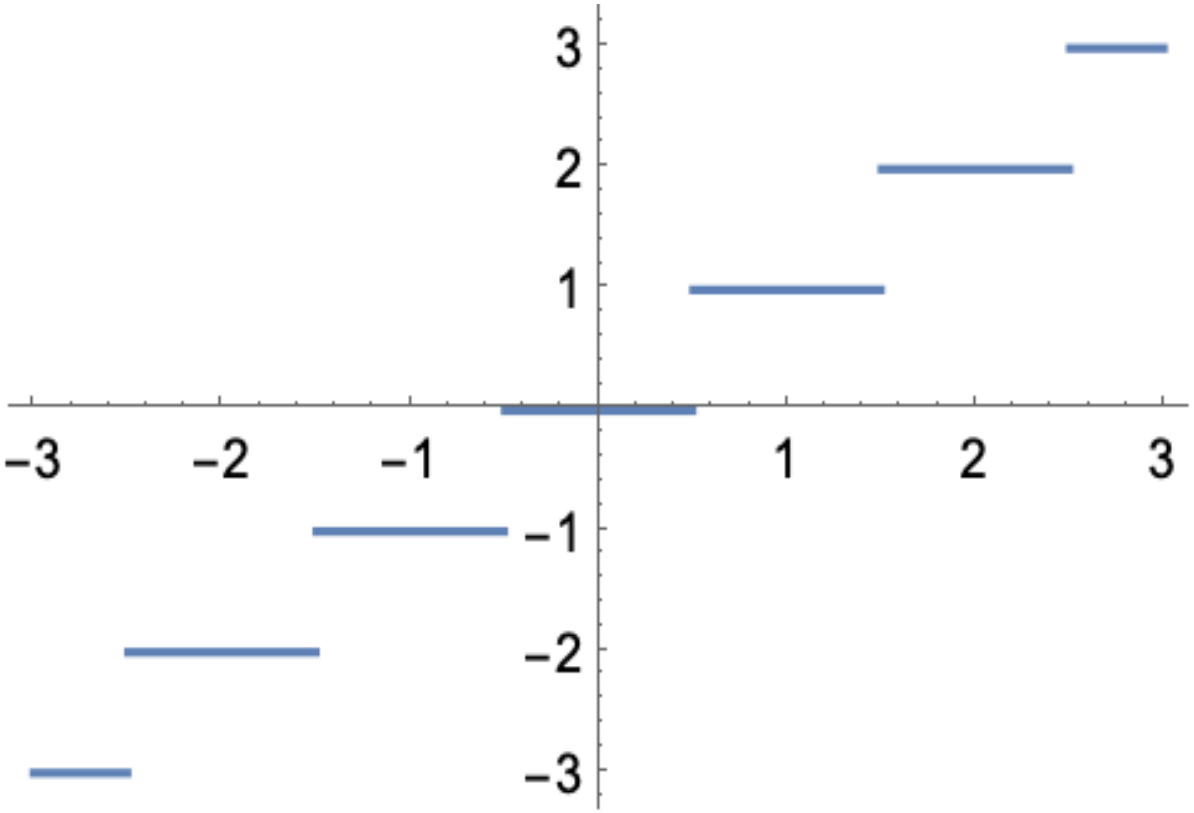


Greatest Integer Function (Floor Function)

Sellable Box Problem

- There are totally 35 cup cakes to be put into boxes of 8 each. How many boxes of cup cakes can be sold?

```
int x = 35 / 8;
```



Integer Rounding Function

Rounding Problem

Round your score to the nearest integer using rounding function.

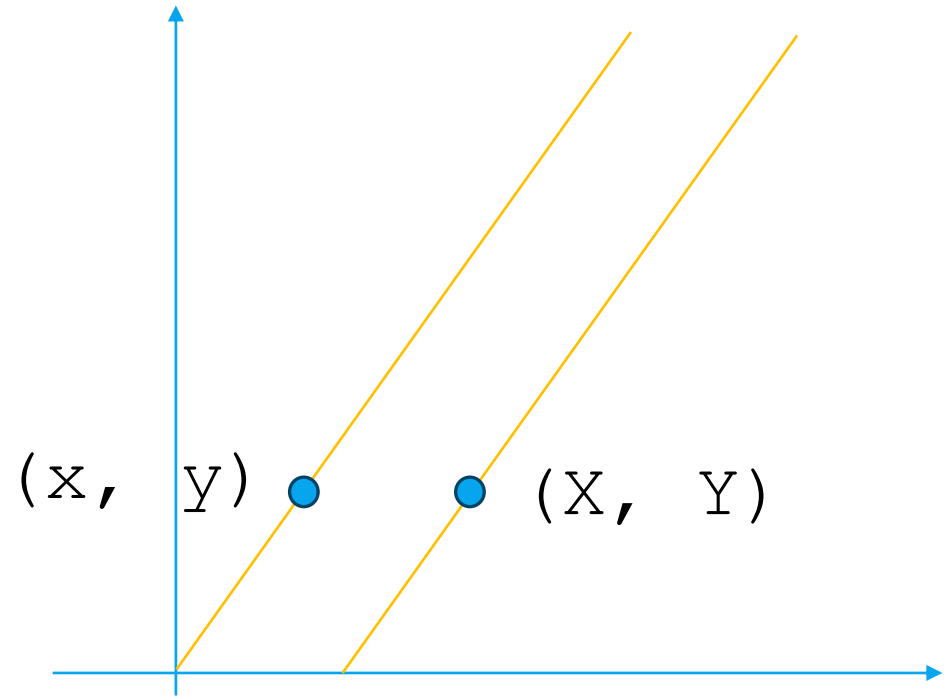
```
int x = (int)(score + 0.5);
```

Functional Transformation

- Translation
- Scaling
- Mirror
- Rotation

X-Y Method (Translation)

$$y = f(x) = 2x$$



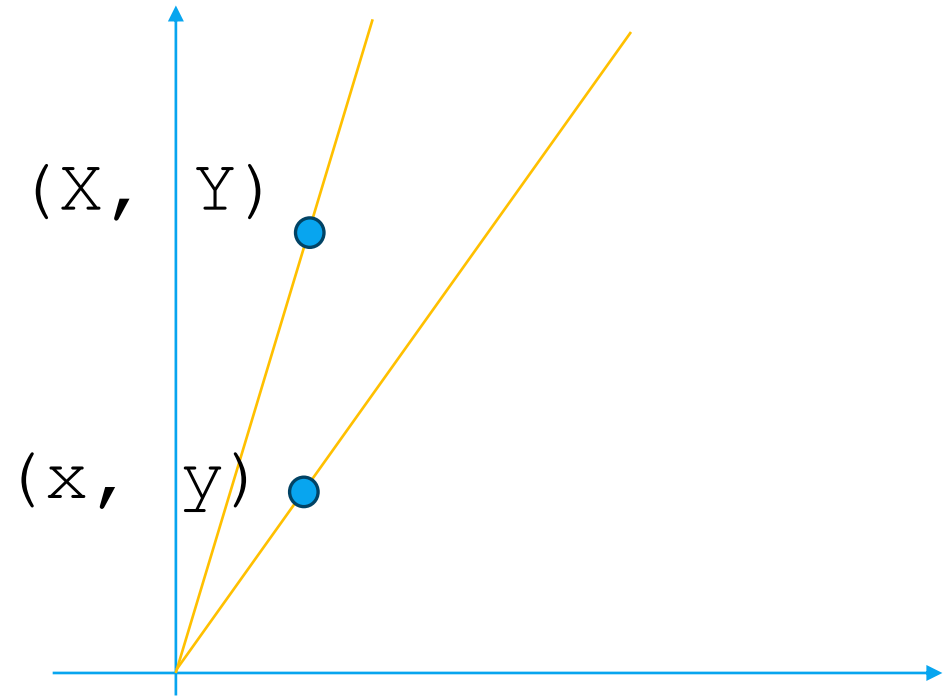
$$X = x + a$$
$$Y = y$$

$$x = X - a$$
$$y = Y$$

$$y = 2x$$
$$Y = 2(X - a)$$

X-Y Method (Scaling)

$y = f(x) = 2x$



$X = x$

$Y = ay$

$x = X$

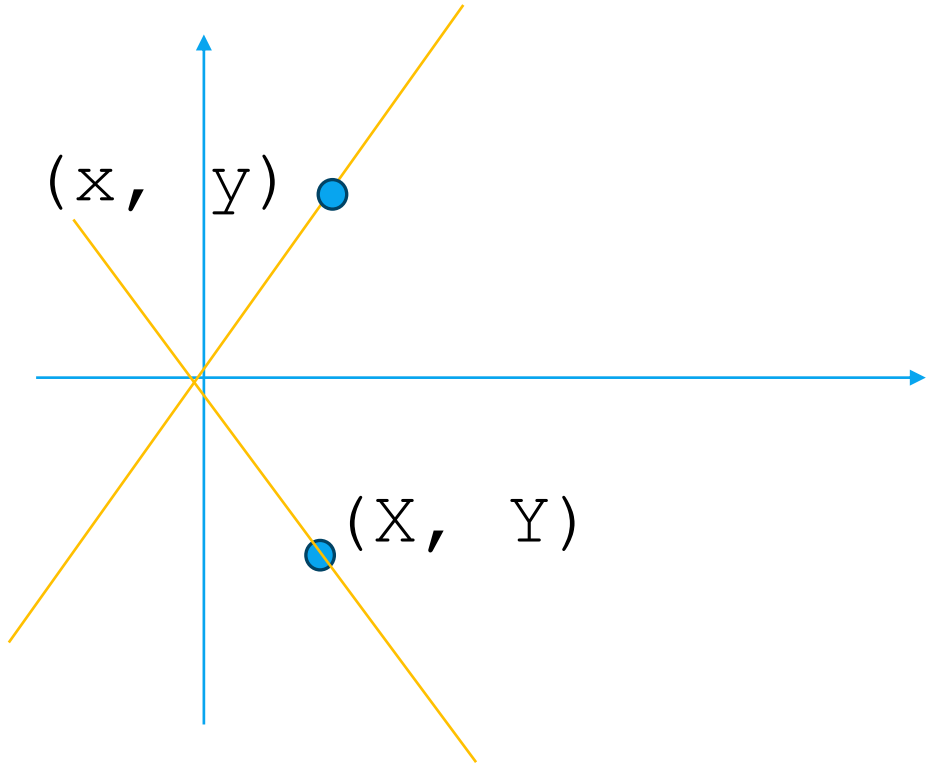
$y = (1/a)Y$

$y = 2\ x$

$Y = 2\ a\ X$

X-Y Method (Mirror)

$$y = f(x) = 2x$$



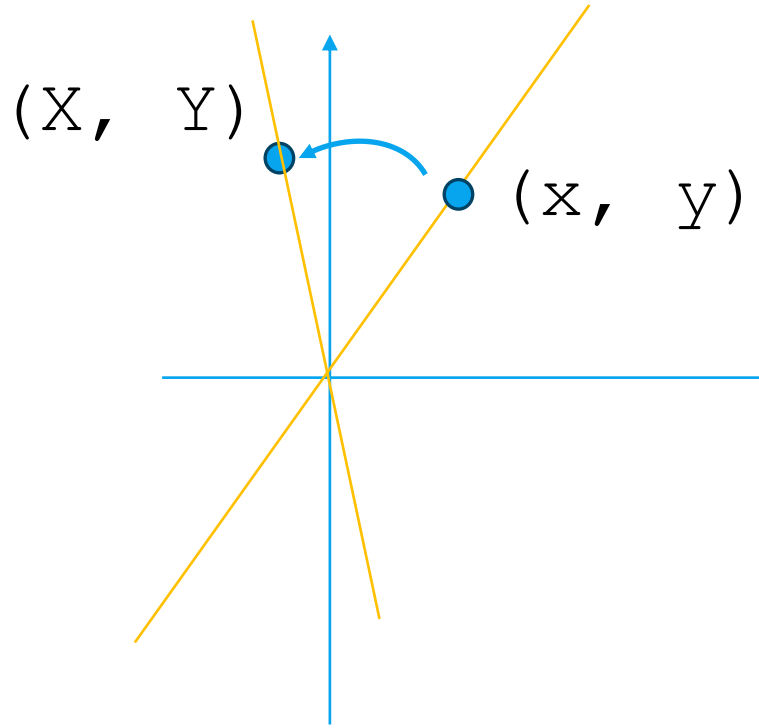
$$X = x$$
$$Y = -y$$

$$x = X$$
$$y = -Y$$

$$y = 2x$$
$$Y = -2X$$

X-Y Method (Rotation)

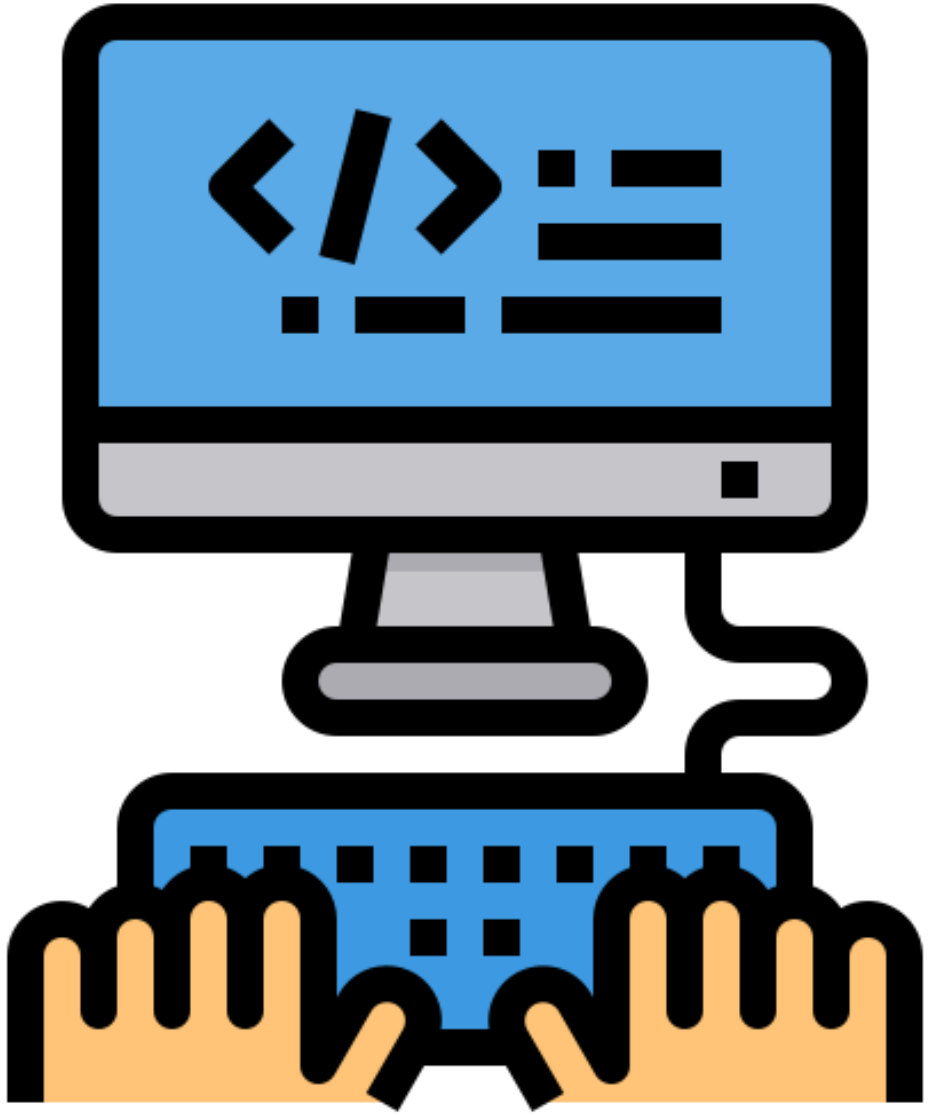
$$y = f(x) = 2x$$



$$X + iY = (x + iy)(\cos\theta + i\sin\theta)$$
$$X = x\cos\theta - y\sin\theta$$
$$Y = y\cos\theta + x\sin\theta$$

$$x + iy = (X + iY)(\cos\theta - i\sin\theta)$$
$$x = X\cos\theta + Y\sin\theta$$
$$y = Y\cos\theta - X\sin\theta$$

$$Y\cos\theta - X\sin\theta = 2\,(X\cos\theta + Y\sin\theta)$$
$$Y = (2\cos\theta + \sin\theta)/(\cos\theta - 2\sin\theta) * X$$



Togglers

Section 2

0-1 Toggler

```
int x = 1;
x = 1 - x; // Toggling Function
```

0-1 Binary Toggler

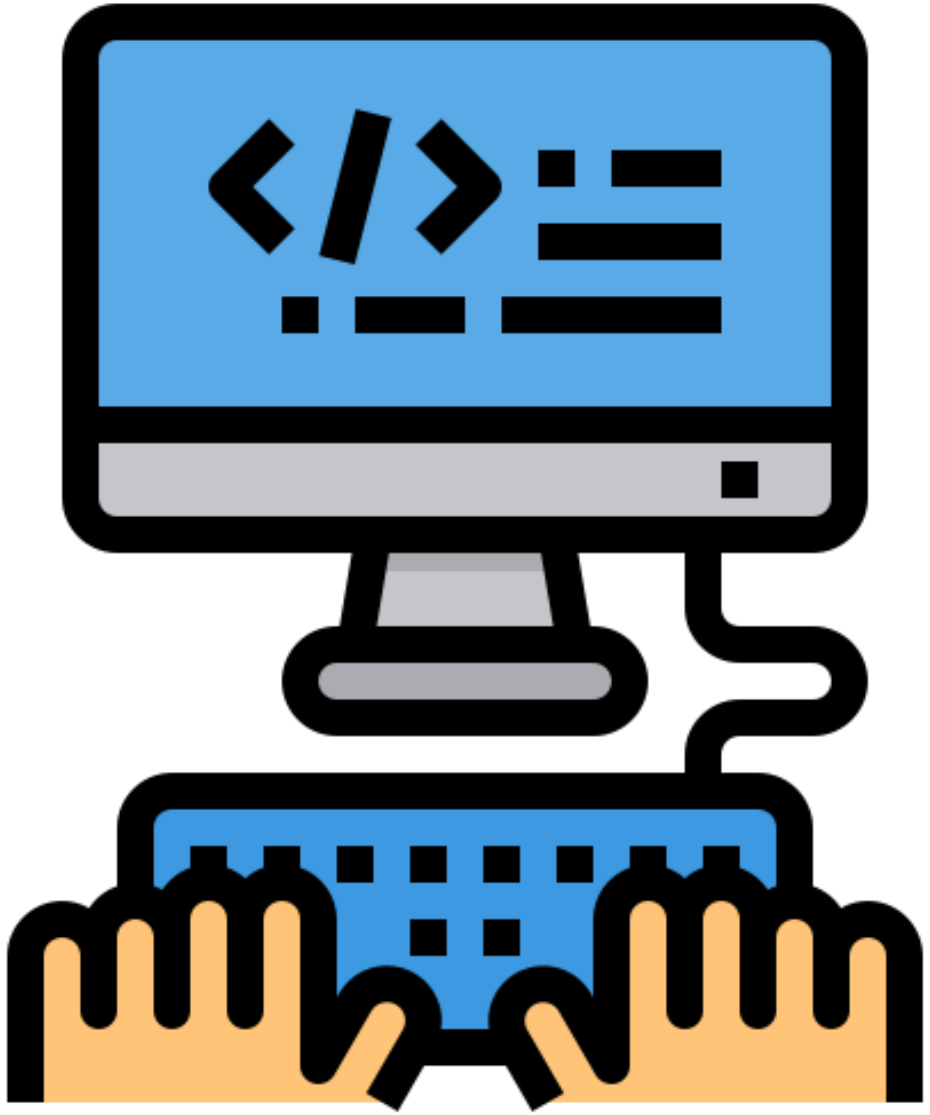
```
2  public class Toggler
3  {
4      int x;
5
6      Toggler(int n){ x=n; }
7      public void toggle(){ x = 1-x; }
8      public String toString() { return ""+x; }
9
10     public static void main(String[] args){
11       Toggler t = new Toggler(1);
12       for (int i=0; i<10; i++){
13           System.out.println(t);
14           t.toggle();
15       }
16     }
17 }
```

3-Way Toggler

```
int x = 0;


x += 1;
x %= 3;
```

3-Way Toggler

```
2  public class TriStateToggler
3  {
4      int x = 0;
5
6      TriStateToggler(){ x= 0; }
7      public void toggle(){ x += 1; x %=3; }
8      public String toString(){ return ""+x; }
9
10     public static void main(String[] args){
11       TriStateToggler t = new TriStateToggler();
12       for (int i=0; i<10; i++){
13           System.out.println(t);
14           t.toggle();
15       }
16     }
17 }
```



Modulus Arithmetic

Section 3

Modulus Arithmetic

- 01/01/2024 is a Monday, how many Fridays in 2024?

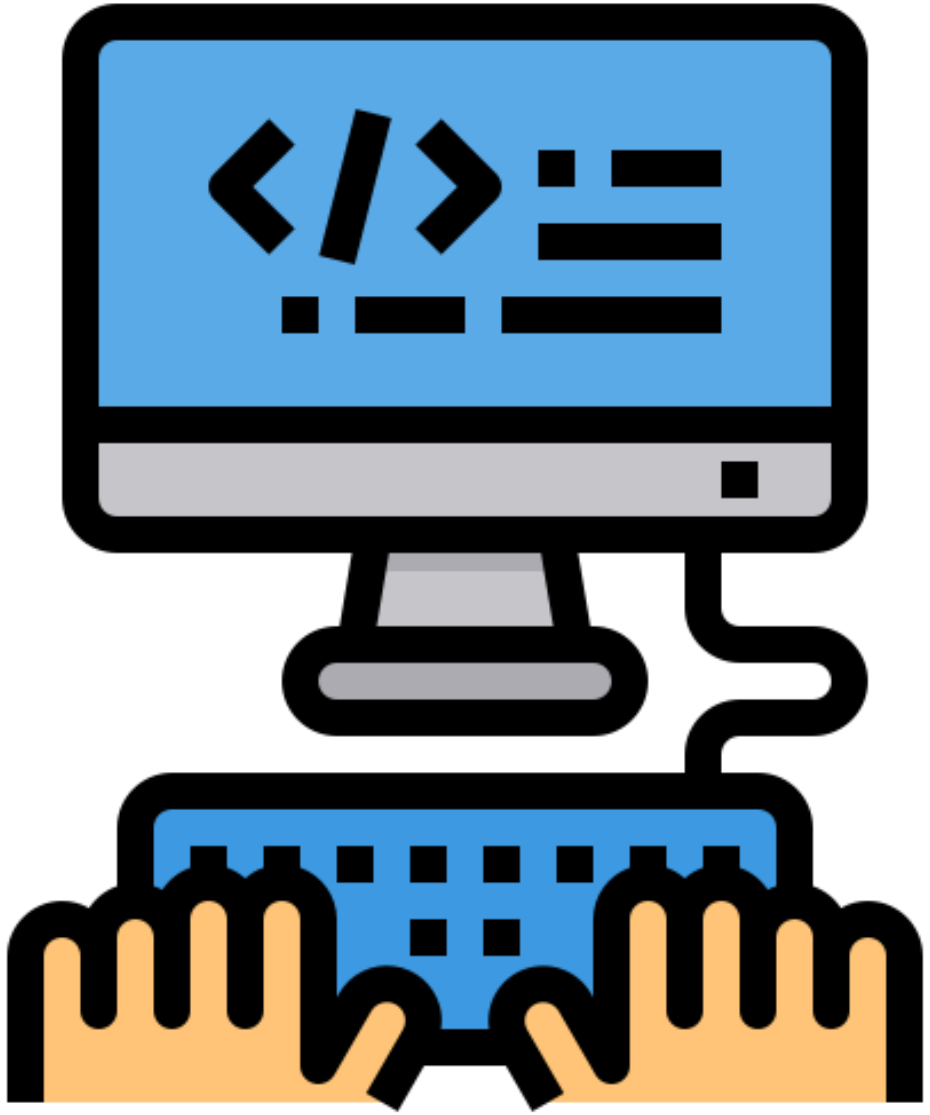
Answer:

There are 366 days in 2024. There are 5 days to have the first Fridays at 01/05/2024. There are 361 days left. 361/7 = 51+ weeks.

There are totally 51+1 Fridays in 2024

USACO Gateway Fridays Problem

- Please refer to USACO Gateway Training site.



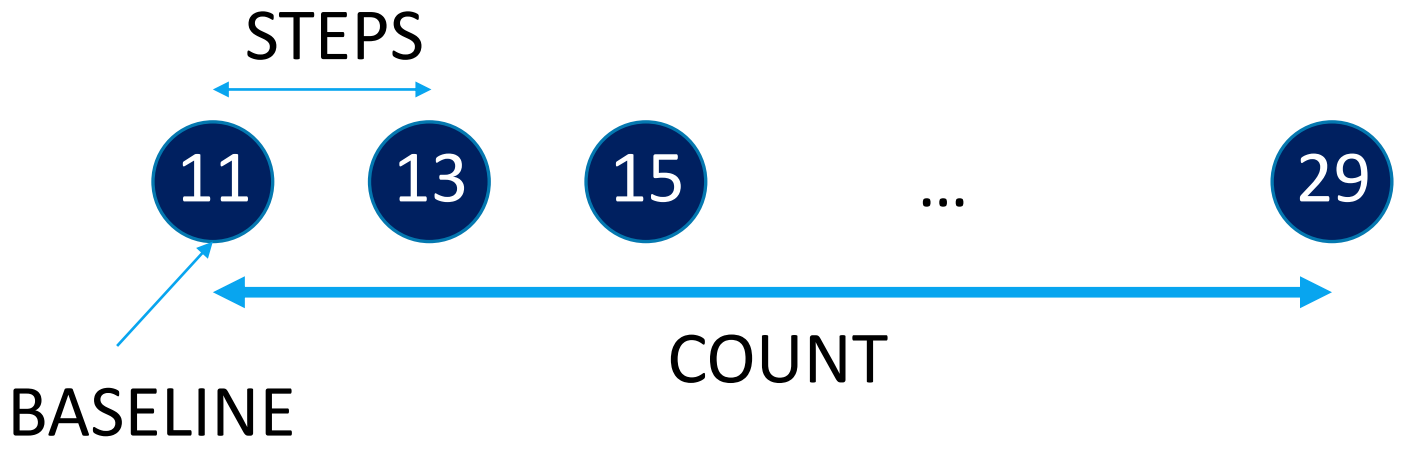
Random Number Generation

Section 4

Review of Random Sample Generation

```
int r = ((int) (Math.random()*COUNT) * STEPS + BASELINE;
```

```
int r = ((int) (Math.random() * 10) * 2 + 11);
```



Generation of a Real Number Range

double r = (Math.random()*range) + BASELINE;

[Baseline, Baseline+range)

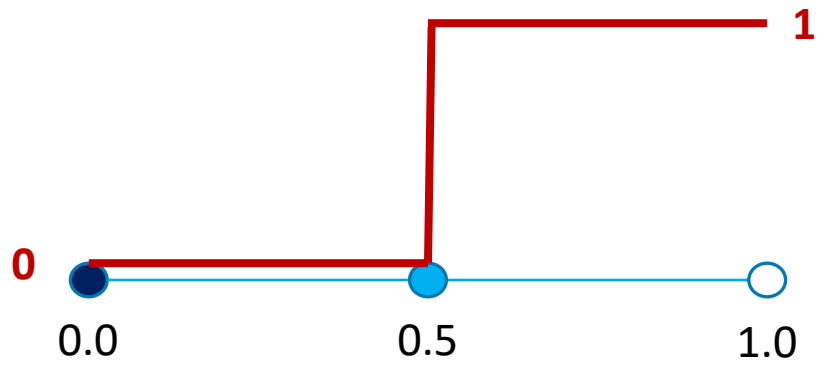
[Baseline, Baseline+range]

```
double y=-999;
      while (y<-Baseline || y>Baseline+range){
         y = Math.random()*(range+0.1)-range/2;
      }
```

Un-biased Randomized Coin (50-50)

Unbiased Random Number Generator:

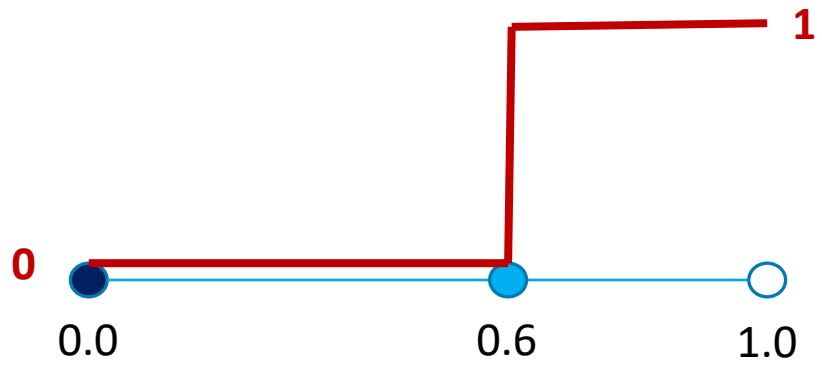
```
double randToss = Math.random();

int die = 1;

if (randToss < 0.5) die = 0;  // preset-else

// Think about it, you do not need the else-part.

// another way to write it.

int ide = (randToss<0.5) ? 0 : 1;

// conditional expression.

//(coming lecture in this chapter)
```

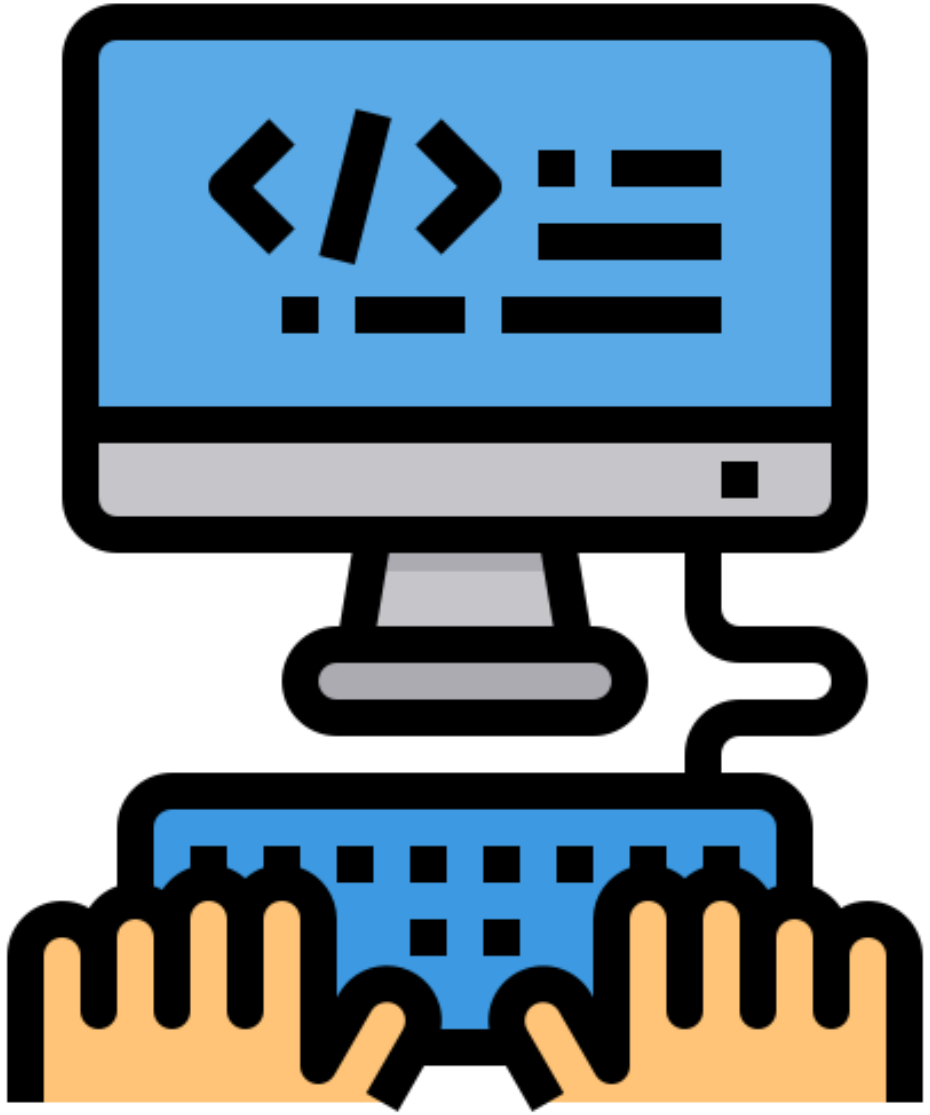


Biased Randomized Coin (60% - 0 (Tail))

Unbiased Random Number Generator:

```
double randToss = Math.random();

int die = 1;

if (randToss < 0.6) die = 0;  // preset-else

// Think about it, you do not need the else-part.

// another way to write it.

int ide = (randToss<0.6) ? 0 : 1;

// conditional expression.

//(coming lecture in this chapter)
```





Successive Division

Section 5

Sum of Digits

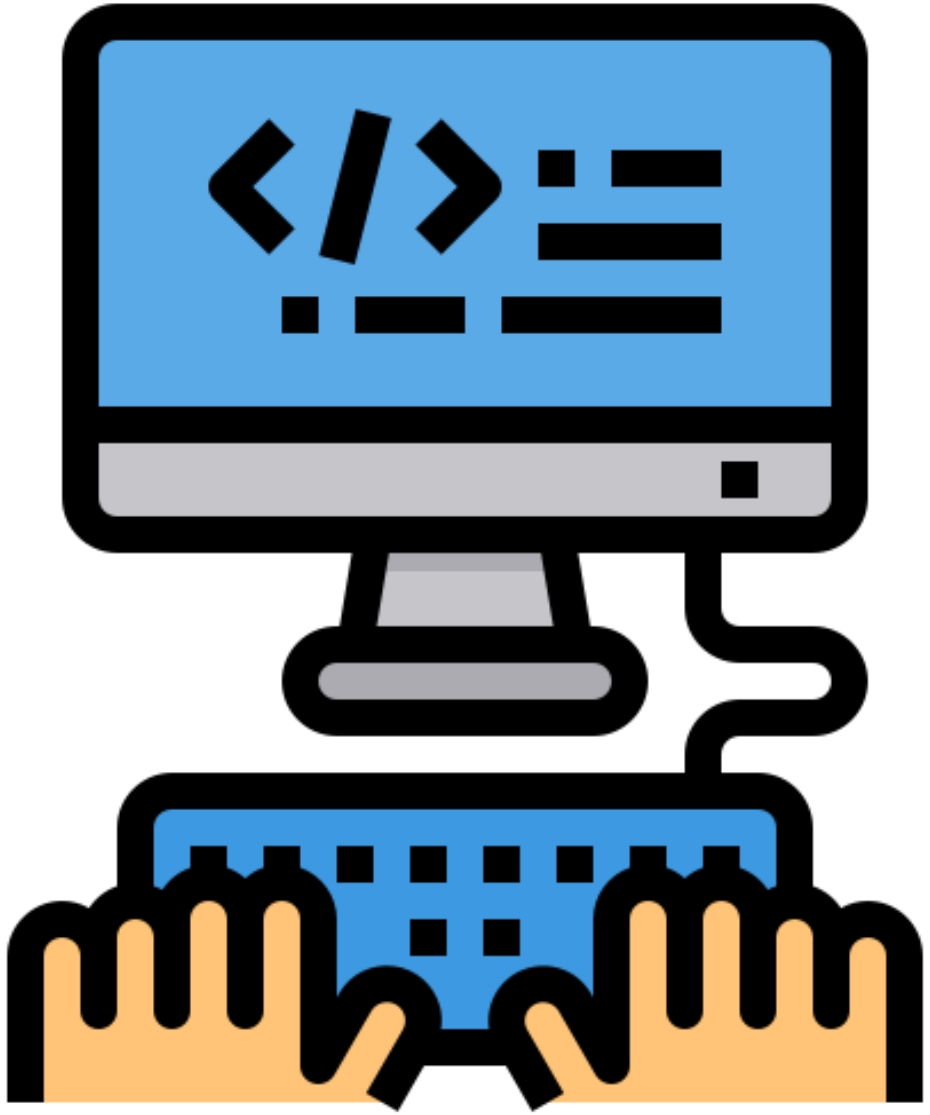
```
public static int sumOfDigits(int n){
    int s = 0;
    while (n>0){
        int d = n%10;
        s += d;
        n /=10;
    }
    return s;
}
```

Reverse of Integer

```
public static int reverse(int n){
    int r = 0;
    while (n>0){
        r = r*10 + n%10;
        n /=10;
     }
    return r;
}
```

Decimal to Binary

```
public static String toBinary(int n){
    String b ="";
    while (n>0){
        int d = n %2;
        b = d + b;
        n /=2;
     }
    return b;
}
```



Indefinite Loop

Section 6

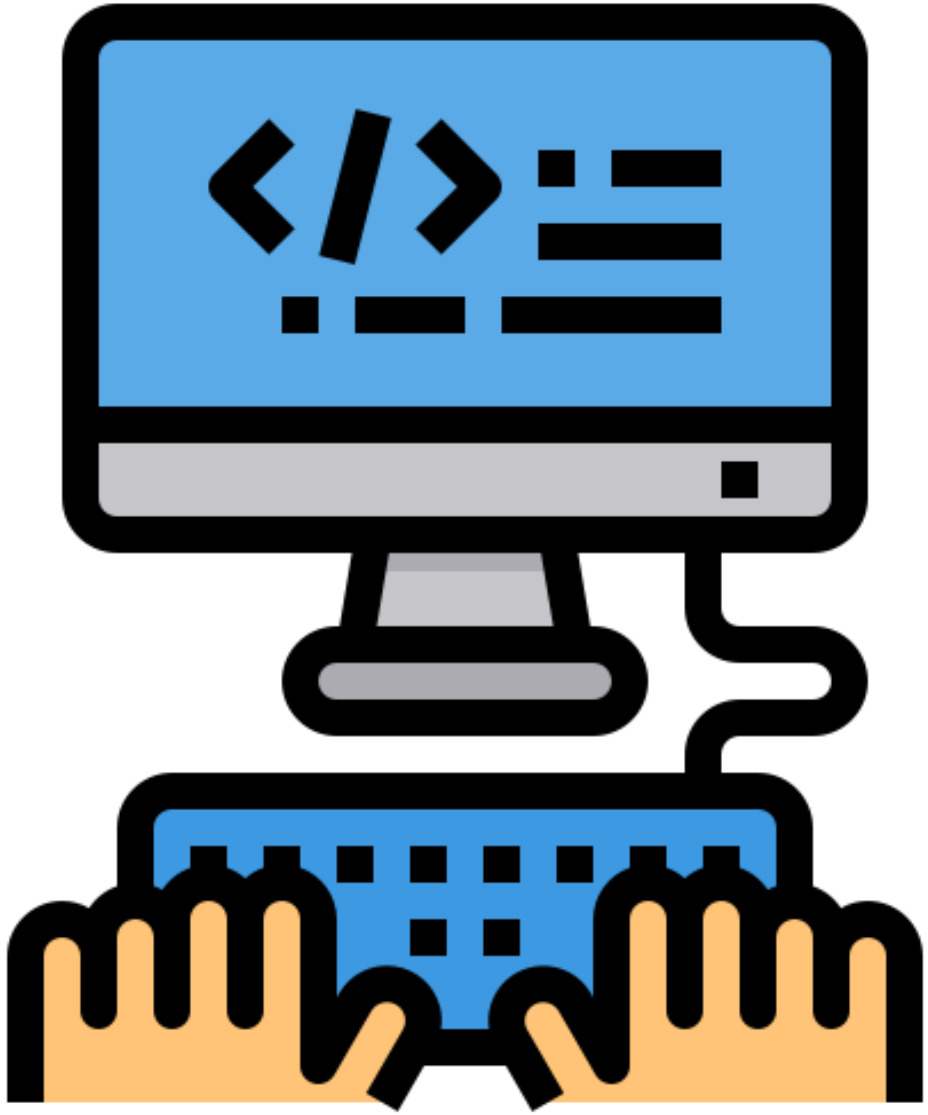
Continue Condition and Exit Condition

```
Continue Condition A:
while (A){
      // …
}

// Post-Condition –A (Exit condition)
```

Use Post-condition to Design Loop

- The post condition will guarantee the result of the loop operation.
- The post condition is the negation of the continue condition

```
1
2  public class RandomInput
3  {
4      public static void main(String[] args){
5          int c = 0;
6
7          int dice = (int)(Math.random()*6)+1;
8          while (dice != 6){
9              c++;
10             dice = (int)(Math.random()*6)+1;
11          }
12         // exit condition: dice == 6
13         System.out.printf("%d toss before a 6\n", c);
14     }
15 }
```



One-Dimensional Traversal

Section 7

Window of 2, Window of 3 Traversal

- Count how many “ab” pattern in a string?

```java
public class CountAB
{
    static String a = "abccabkdjskrfsabbadskajbabbbababaab";

    public static int countAB(String a){
        int c =0;
        for (int i=0; i<a.length()-1; i++){
            if (a.substring(i, i+2).equals("ab")) c++;
         }
        return c;
    }

    public static void main(String[] args){
      System.out.printf("CountAB(a)=%d\n", countAB(a));
    }
}
```

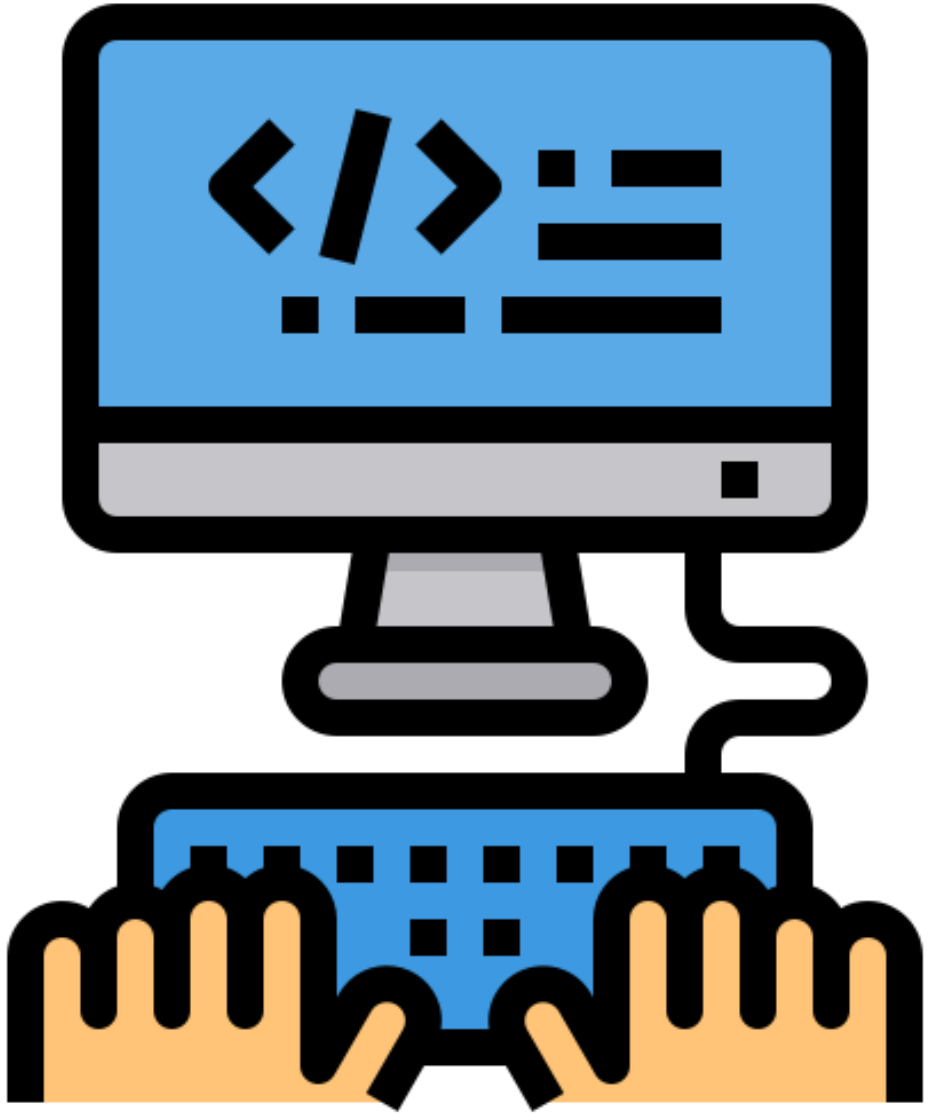
Pattern Replacement

- Replacement of “ab” substring with “cdf”.

```java
public class ReplacementOfAB
{
    static String s = "ab__ab_abb__aabb__ab";

    public static String replace(String s, String pat, String re){
        String r = "";
        for (int i=0; i<s.length(); ){
          if (i<s.length()-pat.length()+1 && s.substring(i, i+pat.length()).equals(pat)){
              r += re;
              i += pat.length();
          }
          else{
              r += s.charAt(i);
              i++;
          }
        }
        return r;
    }

    public static void main(String[] args){
      System.out.printf("String : %s\nReplace: %s\n", s, replace(s, "ab", "cdf"));
    }
}
```

Index Qualifier

- The index needs to be qualified by the s.length()-pat.length()+1



Traversal By Section

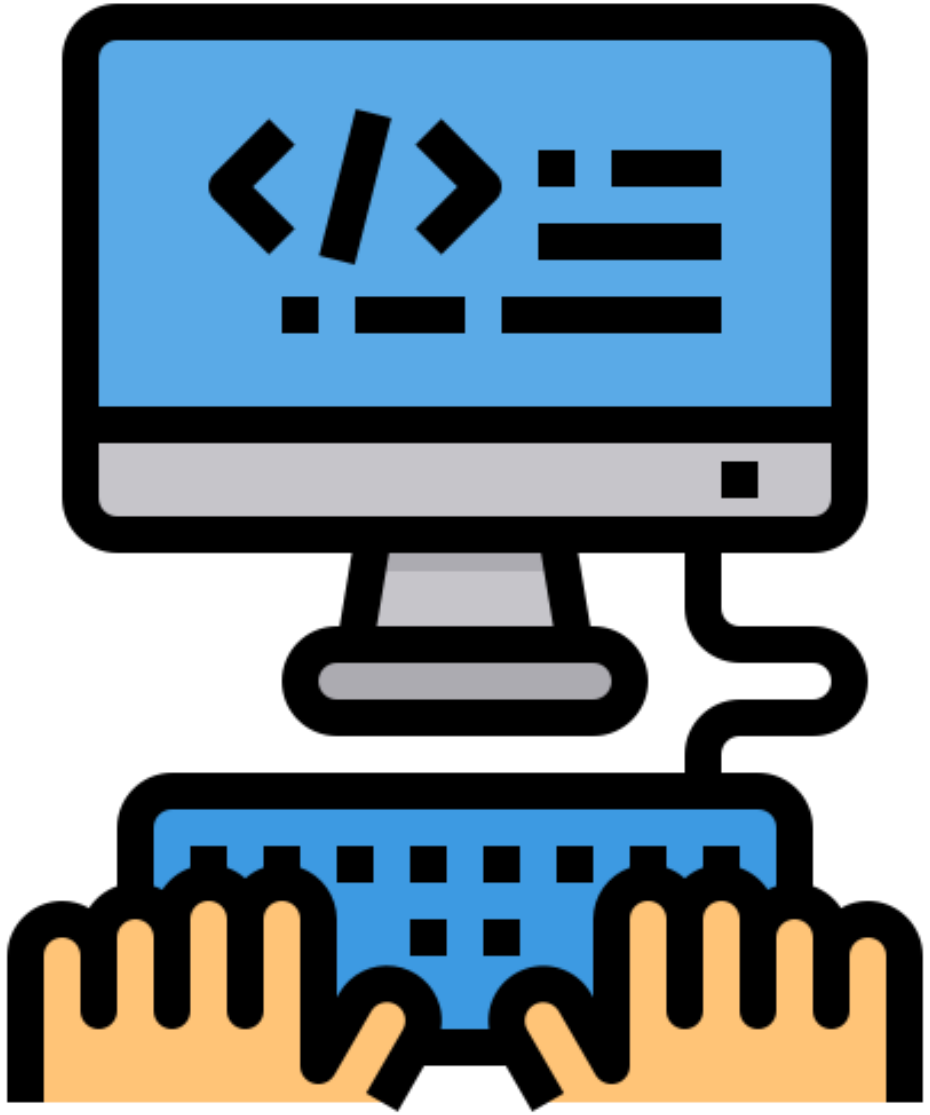
Section 8

Traversal by Section of 3

Given a string, compute a new string by moving the first char to come after the next two chars, so "abc" yields "bca". Repeat this process for each subsequent group of 3 chars, so "abcdef" yields "bcaefd". Ignore any group of fewer than 3 chars at the end.

Traversal by Section of 3

```
static String s = "abcdefghijklmnopqr";

public static String oneTwo(String s){
    String r="";
    for (int i=0; i<s.length()/3*3; i+= 3){
        String three = s.substring(i+1, i+3)+s.charAt(i);
        r += three;
     }
    return r;
}
```



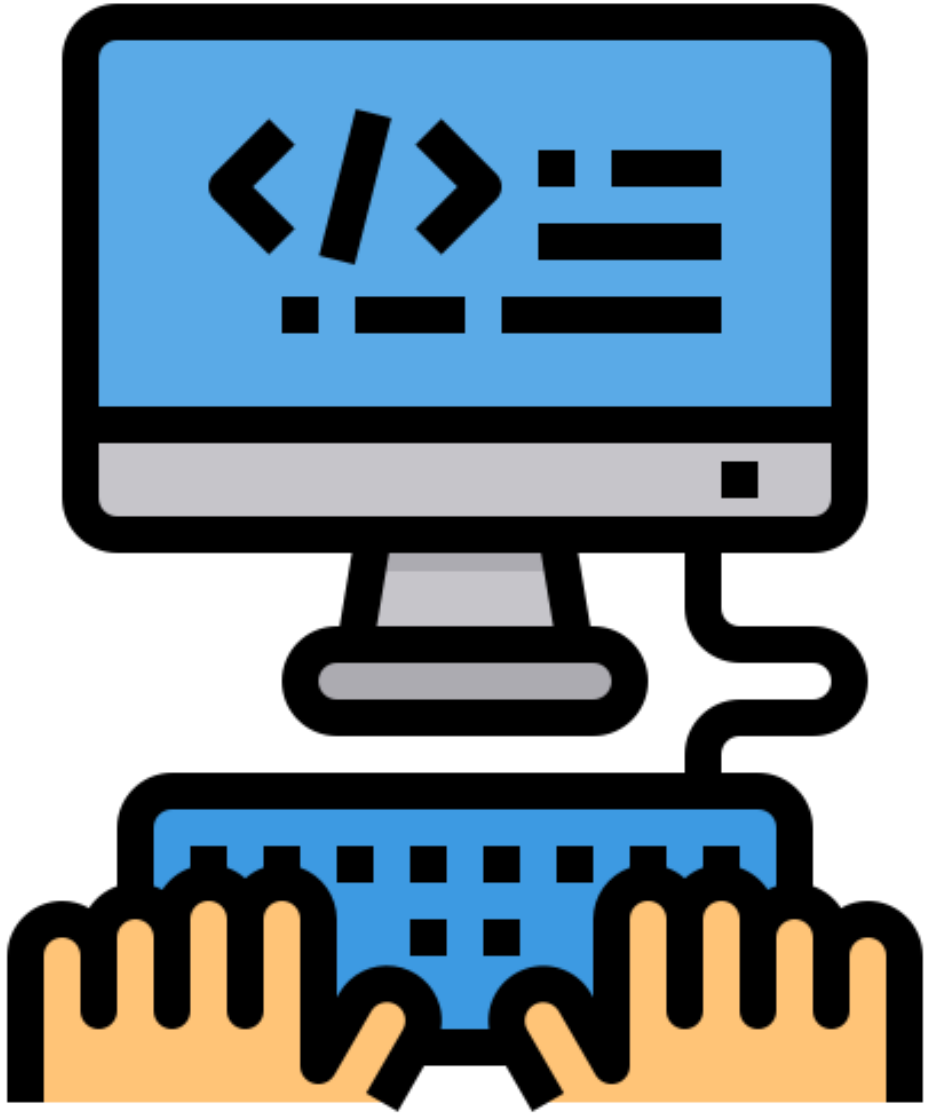
Top 3

Section 9

Use of ArrayList

- Use arraylist to keep track of top 3 numbers.
- Insert each number at the right location and then remove the tail.
- Maintain the arraylist's size to be 3.
- The other method is to keep three variables: gold, silver, bronze.

```java
import java.util.*;
public class Top3
{
    static int[] a = {9, 2, 1, 8, 7, 6, 4, 5, 3};
    static ArrayList<Integer> t = new ArrayList<Integer>(
       Arrays.asList(new Integer[]{0, 0, 0})
    );

    public static void keepTop3(ArrayList<Integer> t, int[] a){
      for (int x: a){
          int d =0;
          while (d<t.size() && t.get(d)>=x) d++;
          t.add(d, x);
          t.remove(t.size()-1);
        }
    }

    public static void main(String[] args){
       keepTop3(t, a);
       System.out.printf("Top 3(a) = %s\n", t);
    }
}
```



All Even/All Odd, Strictly Increasing

Section 10

All Even

```
public static boolean allEven(int[] a){
    for (int i=0;i<a.length;i++){
        if(a[i]%2==1) return false;
    }
    return true;
}
```

Has Even

```
public static boolean hasEven(int[] a){
    for (int i=0; i<a.length; i++){
        if (a[i]%2==0) return true;
     }
    return false;
}
```

Strictly Increasing/Increasing

```
static int[] a={1, 3, 4, 5, 6, 6, 7, 8, 9};

public static boolean isStrictlyIncreasing(int[] a){
    for (int i=0; i<a.length-1; i++){
        if (a[i+1]<=a[i]) return false;
      }
    return true;
 }

public static boolean isIncreasing(int[] a){
    for (int i=0; i<a.length-1; i++){
         if (a[i+1]<a[i]) return false;
      }
    return true;
 }
```