Cosas a tener en cuenta prog 1:

* El primer paso siempre es descomprimir y entender que me pide
* Segundo paso buscar ini y fin de los arr y según lo que me pida va a ser función booleana y int
* Ini y fin pueden ser alterados según sus separadores por ejemplo buscar ini(i<MAX && !mayus) fin(f<MAX && mayus). También si tengo por ejemplo tres variantes de separador (‘,’ ’ ’ ‘.’) defino tres separadores distintos y (ini<MAX) && (mat[ini]==S1 || mat[ini]==S2)

(fin<MAX) && (mat[]ini !=S1 && mat[ini]!=S2)

* Si agrego ósea que hago corrimiento der actualizo fin ++, y si elimino actualizo ini=fin en algunos casos es asi y en otros no
* RECORRER SEC Y SI UN NUM DE SEC COINCIDE CON UNO DE SEC ES TRUE, **POR CADA ELEMENTO DE LA SEC RECORRE TODO EL ARR**:
* fin = buscar\_fin(inicio);
* if (buscar\_produc(inicio, fin) && cant > 0) {
* corrimiento\_der(fin, produc);
* cant--;
* fin++;
* }
* }
* }

Cant es la cantidad de veces que puedo agregar por eso cant - -

Luego muestro cuanto es cant porq me pide cuantos quedaron sin insertar

* public static boolean buscar\_produc (int inicio,int fin) {
* boolean aux=false;
* while (inicio<=fin) {
* int t =0;
* while (t<MAX\_t) {
* if (arr\_p[inicio]==arr\_t[t]) {
* aux=true;
* }
* t++;
* }
* inicio++;
* }
* return aux;
* }
* Si da true,verifico la cantidad en este caso ( if(b\_produc && cant>0) )
* Corrimiento der porque inserta (f,produc a insert)arr[f+1]=produc;
* Después de eso cant --, y fin++ ->actualiza por que agrega elemento y suma fin

**SEGUNDA VARIANTE:**

* Me pide que en dos arr busque si toda una secuencia de uno coincide con toda una sec del otro y si es asi entonces aumenta contador ya que tiene que ser consecutiva por cont tiene q ser mayor a 2 si es mayor elimina y actualiza fin si no es igual vuelve contador a cero porq no es consecutiva

    fin=buscar\_fin(arr\_tiempo,ini);

                if (son\_igual(ini,fin,arr\_tiempo,arr\_patron)) {

                    contador++;

                    if (contador>N) {

                        eliminar(ini,fin,arr\_tiempo);

                        fin=ini;

                    }

                } else{

                    contador=0;

                }

            }

public static boolean son\_igual(int ini,int fin, int[]arr\_tiempo,int[]arr\_patron){

        boolean aux = false;

        int ini\_p=0,fin\_p=-1;

            if (ini\_p<MAX) {

                ini\_p=buscar\_inicio(arr\_patron, fin\_p+1);

                fin\_p=buscar\_fin(arr\_patron, ini\_p);

                if (fin\_p-ini\_p+1==fin-ini+1 && comparar(ini,ini\_p,fin,arr\_tiempo,arr\_patron)) {

                    aux=true;

                }

            }

        return aux;

    }

En este caso son similares pero para una sec busca que coincida con una de todo el segundo arr entonces para una misma sec va buscando distintas sec en el segundo arr y si una de esas coincide entonces da true->por eso es while y no if

Son igual es la misma función.

public static boolean buscarIgual(int inicio, int fin, char[] arrPalabras, char[] arrStopwords) {

        int inicioS = 0, finS = -1;

        boolean sies = false;

        while (inicioS < MAX\_S) {

            inicioS = buscarInicioS(finS + 1, arrStopwords);

            if (inicioS < MAX\_S) {

                finS = buscarFinS(inicioS, arrStopwords);

                if ((fin - inicio + 1 == finS - inicioS + 1) && (sonIgual(inicio, fin, arrPalabras, inicioS, finS, arrStopwords))) {

                    sies = true;

                }

            }

        }

        return sies;

    }

* en esta función busca ini y fin del segundo arr, luego compara la longitud y si son iguales y tienen los mismos elementos da verdadero
* comparar compara que todos los elementos coincidan
* public static boolean comparar(int ini, int ini\_p, int fin, int[]arr\_tiempo,int[]arr\_patron){
* boolean aux=false;
* while (ini<=fin && arr\_tiempo[ini]==arr\_patron[ini\_p]) {
* ini++;
* ini\_p++;
* }
* if (ini>fin) {
* aux=true;
* }
* return aux;

**TERCERA VARIANTE:**

 fin = finSec(arr, ini);

                //Procesamiento de la secuencia

                if (estaNumSecOrd(arr, ini, fin, numeroBus) ){

                  invertirSec(arr, ini, fin);

                }

                else{

                  rellenarSecNum(arr, ini, fin, numeroReem);

                }

Si esta num dado invierte la sec y sino esta la rellena con otro num

public static boolean estaNumSecOrd(int[] arr, int ini, int fin, int num){

             int i = ini;

             while (i <= fin && arr[i] < num){

               i++;

             }

              if(i <= fin && arr[i] == num){

                return true;

              }

              else

                  return false;

**otro ejemplo igual:**

  public static boolean buscar\_produc(int ini, int fin,int[]arr\_libros,int num){

        boolean aux=false;

            while (ini<=fin) {

                if (arr\_libros[ini]<=num) {

                    aux=true;

                }

                ini++;

            }

        return aux;

    }

public static void invertirSec(int[] arr, int ini, int fin){

             int i = ini;

             int j = fin;

             while(i < j){

               int aux = arr[i];

               arr[i] = arr[j];

               arr[j] = aux;

               i++;

               j--;

             }

          }

          public static void rellenarSecNum(int[] arr, int ini, int fin, int num){

            for(int i = ini; i <= fin; i++){

              arr[i] = num;

**CUARTA VARIANTE:**

 fin=buscar\_fin(arr\_imagen,inicio);

                sec = fin-inicio+1;

                if (sec>X && son\_igual(arr\_imagen,inicio,fin)) {

                    comprimir(inicio,fin,arr\_imagen,sec);

                    fin=inicio;

En este caso si el largo de la sec supera X y son igual da true->verifica si ini es igual el sig elemento, mientras sea igual sigue iterando->si ini llego al final da true(se rompe si ini no llego a fin y salio porq no son iguales)

En este caso si coincidia tenia que poner el numero que estaba descomprimido y al lado la cantidad en negativo

public static void comprimir(int inicio,int fin, int[]arr\_imagen, int sec){

        for(int i=inicio; i<fin-1; i++){

            corrimiento\_izq(arr\_imagen, inicio);

        }

        arr\_imagen[inicio+1]=sec\*-1;

i<fin-1(porq deja un num). Arr[ini+1]=sec\*-1(le agrega al segundo elemento el num de ocurrencias en neg)

 public static boolean son\_igual(int[] arregloImagen, int inicio, int fin) {

        boolean aux=false;

while (inicio<=fin && arregloImagen[inicio]==arregloImagen[inicio+1]){

                inicio++;

            }

            if (inicio==fin) { //if (inicio>fin) dos arr

            aux=true;

            }

        return aux;

        }

**QUINTA VARIANTE:**

En este caso recorre primero el arr 2 y convierte el elemento en el numero de columna de la mat, busca en ese dia si los tiempos bajaron, si da true significa q si y q cumplio por los tanto a la pos del arr le agrega 0 o elimina y suma los días q si cumplio

  for (int i = 0; i < MAX\_Arr && arr[i] != 0; i++) {

            totalDias++;

            int dia = arr[i] - 1; // Convertir el día de entrenamiento al índice de la matriz

            if (arr\_tiempo(mat[dia])) {

                System.out.println("El día " + arr[i] + " cumplió con el entrenamiento progresivo");

                arr[i] = 0; // Eliminar día del arreglo si se cumplió

                diasCumplidos++;

            } else {

                System.out.println("El día " + arr[i] + " no cumplió con el entrenamiento progresivo");

            }

        }

        if (totalDias > 0) {

            double porcentajeCumplido = (double) diasCumplidos/totalDias;

            if (porcentajeCumplido >= P) {

                System.out.println("cumplio: " + porcentajeCumplido + "|" + diasCumplidos + "|" + totalDias);

            } else {

                System.out.println("no cumplio");

            }

        } else {

            System.out.println("no hay dias asignados para entrenamiento progresivo");

 boolean sec\_decendente = true;

        while (ini<MAXCOL && sec\_decendente) {

            ini=buscar\_ini(mat,fin+1);

            if (ini<MAXCOL) {

                fin=buscar\_fin(mat,ini);

                if(!decrece(ini,fin,mat)){

                    sec\_decendente = false;

Si decrece da true, sigue iterando por todas las sec cuando da false para y devuelve false->es decir una q no coincida corta

 public static boolean decrece(int ini,int fin, int []mat){

        while (ini<fin && mat[ini] >= mat[ini+1]) {

            ini++;

        }

        if (ini==fin) {

            return true;

        }

        return false;

**SEXTA VARIANTE:**

**Lo** q hace esto busca en un arr dado el primer elemento le resta uno para que sea compatible con la col de la mat y el segundo for hace algo similar pero con otro arr

 for(int i=0; i<MAX\_Arr && arr1[i]!=SEPARADOR; i++){

                int j = arr1[i]-1;

                promedio = calcular\_dia(matriz[j]);

                System.out.println("El mes "+arr1[i]+" su prom es: "+promedio);

            }

            for(int i=0; i<MAX\_Arr && arr2[i]!=SEPARADOR; i++){

                int j2 = arr2[i]-1;

                if (calcular\_mes(matriz[j2],X)) {

                    System.out.println("cumplio");

                }

                else{

                    System.out.println("no cumplio");

                }

            }

Busca cada num mayor en cada sec, los suma a todos y los divide según la cantidad de sec=días y ese es el total

fin=buscar\_fin(arr,ini);

                int num = buscar\_mayor(ini,fin,arr);

                num\_mayor+=num;

                dias++;

            }

        }

        int T = num\_mayor/dias;

        return T;

public static int buscar\_mayor(int ini,int fin,int[]arr){

        int mayor=0;

        while (ini<fin) {

            if (arr[ini]>mayor) {

                mayor=arr[ini];

            }

            ini++;

En el segundo for llama a calcular mes donde busca ini y fin de nuevo en la mat ya q este busca en la ultima columna, suma todos los elementos de la sec y los divide por su long, si ese total supera num dado X entonces devuelve true ósea q cumple

fin=buscar\_fin(arr2,ini);

                if (!es\_mayor(ini,fin,arr2,X)) {

                    return false;

                }

            }

        }

        return true;

public static boolean es\_mayor(int ini,int fin, int[]arr2, int X){

        int sec=0,sum=0;

        sec = fin-ini+1;

        while (ini<=fin) {

            sum+=arr2[ini];

            ini++;

        }

        int total = sum/sec;

        if (total>X) {

            return true;

        }else{

            return false;

        }

**SEPTIMA VARIANTE:**

En estos procedimientos lo q hace es si ini de sec es mayus entonces va a encriptar, una vez que encripta actualiza fin la cantidad de veces y aumenta el contador que luego lo devuelve a otra función que acumula y suma total

fin=buscar\_fin(arr,ini);

                if (Es\_mayus(arr[ini])) {

                    encriptado = encriptacion(ini, fin, arr);

                    fin+=encriptado;//PREGUNTARR

                    cantT++;

**P**rimero invierte como ya lo vimos y luego duplica las vocales por cada que duplica aumenta cont y devuelve a cant q acumula

 public static int encriptacion(int ini,int fin,char[]arr){

        int cant=0;

            invertir(ini,fin,arr);

            cant+=duplicar\_vocales(ini,fin,arr);

            return cant;

        }

    public static int duplicar\_vocales(int ini, int fin,char []arr) {

        int cont=0;

        while (ini <= fin) {

            if (es\_vocal(arr[ini])) {

                desplazar\_derecha(arr,ini);

                cont++;

                ini++;

                fin++;//mueve doble para no duplicar el duplicado

            }

            ini++;

        }

        return cont;

**METODOS A TENER EN CUENTA**

 public static boolean es\_vocal(char c) {

        return c == 'a' || c == 'e' || c == 'i' || c == 'o' || c == 'u';

    }

    public static boolean Es\_mayus(char arr) {

        return arr >= 'A' && arr <= 'Z';

    }