

BITACORA DE EJERCICIOS

Nicolas Caro Ñustez 30000049057



Contenido

[Ejercicio 1 2](#_Toc35037835)

[Ejercicio 2 3](#_Toc35037836)

[Ejercicio 3 5](#_Toc35037837)

[Ejercicio 4 8](#_Toc35037838)

[Ejercicio 5 12](#_Toc35037839)

[Ejercicio 6 14](#_Toc35037840)

[Ejercicio 7 16](#_Toc35037841)

[Ejercicio 8 17](#_Toc35037842)

[Ejercicio 9 20](#_Toc35037843)

[Ejercicio 10 23](#_Toc35037844)

[Ejercicio 11 25](#_Toc35037845)

[Ejercicio 12 27](#_Toc35037846)

# Ejercicio 1

Imagen que contiene captura de pantalla

Descripción generada automáticamente

public class Principal {

public static void main(String[] args) {

try{

Scanner sc = new Scanner(System.in);

DecimalFormat df = new DecimalFormat("#.000");// para usar unicamente los ultimos 3 decimales

int d = sc.nextInt();// distancia

int vm = sc.nextInt();// velocidad media

System.out.println(df.format((double)(d\*vm)/12));

}catch(InputMismatchException a){

System.out.println("Por favor solo digite numeros");

}

}

}

# Ejercicio 2

Imagen que contiene captura de pantalla

Descripción generada automáticamente

public class Principal {

public static void main(String args[]) {

try{

int a , b, x ;

Scanner lector = new Scanner(System.in);

System.out.println("Ingrese el primer digito: ");

a=lector.nextInt();

System.out.println("Ingrese el segundo digito: ");

b=lector.nextInt();

x=a+b;

System.out.println("X"+" "+"="+" "+x);

}catch(InputMismatchException ab){

System.out.println("Presentation Error");

}

}

}

# Ejercicio 3

Imagen que contiene captura de pantalla

Descripción generada automáticamente

public class Pirncipal {

public static void main(String[] args) {

try {

Scanner lector = new Scanner(System.in);

System.out.println("Digite el salario");

double salario = lector.nextFloat();

if (salario >= 0 && salario <= 400.00) {

double operacion = salario \* 0.15;

double salarioN = salario + operacion;

System.out.println("Nuevo Salario:" + salarioN);

System.out.println("Dinero Ganado: " + operacion);

System.out.println("En Porcentaje: " + "15%");

} else {

if (salario > 400.01 && salario < 800.00) {

double operacion = salario \* 0.12;

double salarioN = salario + operacion;

System.out.println("Nuevo Salario:" + salarioN);

System.out.println("Dinero Ganado: " + operacion);

System.out.println("En Porcentaje: " + "12%");

} else {

if (salario >= 800.01 && salario <= 1200) {

double operacion = salario \* 0.1;

double salarioN = salario + operacion;

System.out.println("Nuevo Salario:" + salarioN);

System.out.println("Dinero Ganado: " + operacion);

System.out.println("En Porcentaje: " + "10%");

} else {

if (salario >= 1200.01 && salario <= 2000) {

double operacion = salario \* 0.07;

double salarioN = salario + operacion;

System.out.println("Nuevo Salario:" + salarioN);

System.out.println("Dinero Ganado: " + operacion);

System.out.println("En Porcentaje: " + "7%");

} else {

if (salario > 2000) {

double operacion = salario \* 0.04;

double salarioN = salario + operacion;

System.out.println("Nuevo Salario:" + salarioN);

System.out.println("Dinero Ganado: " + operacion);

System.out.println("En Porcentaje: " + "4%");

}

}

}

}

}

} catch (InputMismatchException ab) {

System.out.println("Digite caracteres validos");

}

}

}

# Ejercicio 4

Imagen que contiene captura de pantalla

Descripción generada automáticamente

# 

public class Principal {

public static void main(String[] args) {

Scanner a = new Scanner(System.in);

String animal = a.next();

String tipo=a.next();

String comida = a.next();

if(animal.equals("vertebrado")){

if(tipo.equals("ave")){

if(comida.equals("carnivoro")){

System.out.println("aguila");

}else{

if(comida.equals("onivoro")){

System.out.println("pomba");

}

}

}else{

if(tipo.equals("mamifero")){

if(comida.equals("onivoro")){

System.out.println("homem");

}else{

if(comida.equals("herbivoro")){

System.out.println("vaca");

}

}

}

}

}else{

if(animal.equals("invertebrado")){

if(tipo.equals(animal)){

if(comida.equals(animal)){

}else{

if(comida.equals(animal)){

}

}

}else{

if(tipo.equals("insecto")){

if(comida.equals("hematofago")){

System.out.println("pulga");

}else{

if(comida.equals("hervivoro")){

System.out.println("lagarta");

}

}

}

else{

if(tipo.equals("anelido")){

if(comida.equals("hematofago")){

System.out.println("sangessuga");

}else{

if(comida.equals("onivoro")){

System.out.println("minhoca");

}

}

}

}

}

}else{

System.out.println("No existe animal");

}

}

}

}

# Ejercicio 5

Imagen que contiene captura de pantalla

Descripción generada automáticamente

public class Principal {

public static void main(String[] args) {

Locale.setDefault(new Locale("en", "US"));

Scanner sc = new Scanner(System.in);

int t = Integer.parseInt(sc.nextLine());

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

String sheldon = sc.next();

String raj = sc.next();

System.out.printf("Caso #%d: ",i+1);

//empate: "De novo!".

if(sheldon.equals(raj)) System.out.println("De novo!");

//Sheldon vence: "Bazinga!"

else if(sheldon.equals("tesoura") && (raj.equals("papel") || raj.equals("lagarto"))) System.out.println("Bazinga!");

else if(sheldon.equals("papel") && (raj.equals("pedra") || raj.equals("Spock"))) System.out.println("Bazinga!");

else if(sheldon.equals("pedra") && (raj.equals("lagarto") || raj.equals("tesoura"))) System.out.println("Bazinga!");

else if(sheldon.equals("Spock") && (raj.equals("tesoura") || raj.equals("pedra"))) System.out.println("Bazinga!");

else if(sheldon.equals("lagarto") && (raj.equals("Spock") || raj.equals("papel"))) System.out.println("Bazinga!");

//Raj vence: "Raj trapaceou!"

else System.out.println("Raj trapaceou!");

}

sc.close();

}

}

# Ejercicio 6

Imagen que contiene captura de pantalla

Descripción generada automáticamente

public static void main(String[] args) {

Scanner lector = new Scanner(System.in);

System.out.println("Digite el primer valor: ");

int a = lector.nextInt();

System.out.println("Digite el segundo valor: ");

int b = lector.nextInt();

int q, r;

if (a >= 0) {

q = a / b;

r = a % b;

} else {

int x = 0, y = 0;

if (b < 0) {

x = b \* -1;

} else {

x = b;

}

for (r = 0; r < x; r++) {

y = a - r;

if (y % b == 0) {

break;

}

}

q = y / b;

}

System.out.println("La respuesta es: "+q + " " + r);

}

# Ejercicio 7

Imagen que contiene captura de pantalla

Descripción generada automáticamente

public static void main(String[] args) {

Scanner lector = new Scanner(System.in);

System.out.println("Digite el valor: ");

int a = lector.nextInt();

int b[] = new int[a];

int c = 0;

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

b[i] = lector.nextInt();

}

if (a > 2) {

for (int i = 2; i < a; i++) {

if ((b[i - 2] > b[i - 1] && b[i - 1] < b[i]) || (b[i - 2] < b[i - 1] && b[i - 1] > b[i])) {

c = 1;

} else {

c = 0;

break;

}

}

} else {

if (b[0] != b[1]) {

c = 1;

}

}

System.out.println("La respuesta es: " + c);

}

# Ejercicio 8

Internship

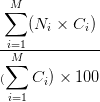
By Ricardo Oliveira, UFPR BR Brazil

**Timelimit: 1**

*Googlbook* is a famous IT company that opened an office in your town this year! Also, *Googlbook* has just offered interviews to an internship position in the company!

To be interviewed, you need to send some personal information to the company, that will be used to decide who will earn the position. You sent all information they need except one: your API (Academic Performance Index). To get things worse, *Student’s Portal*, the system that provide your API, is not working!

Fortunately, you remember all the grades you got in all **M** subjects you coursed, as well their workloads. You also remember how the API is calculated:



, where **N1**, **N2**, ..., **NM** are your grades in each subject, and **C1**, **C2**, ..., **CM** are the workload of the respective subjects.

Given the grades you got and the workload of each subject, determine your API, so you can send it to Googlbook as soon as possible!

**Input**

The input contains several test cases. The first line of each test case contains integer **M** (1 ≤ **M** ≤ 40), the number of subjects you coursed. Each of the next **M** lines describe a subject. Each line contains two integers **Ni** and **Ci** (0 ≤ **Ni** ≤ 100, 30 ≤ **Ci** ≤ 120), indicating the grade you got in that subject and its workload, respectively.

The input ends with end-of-file (EOF).

**Output**

For each test case, print a line containing your API. Round and print it with exactly 4 decimal places.

| **Input Sample** | **Output Sample** |
| --- | --- |
| 3 70 60 90 60 80 120 | 0.8000 |

import java.text.DecimalFormat;

import java.util.Scanner;

/\*\*

\*

\* @author nico\_

\*/

//Internship

public class Principal {

public static void main(String[] args) {

System.out.println("Digite los valores: ");

Scanner lector = new Scanner(System.in);

DecimalFormat e = new DecimalFormat("0.0000");

int a;

while (lector.hasNext()) {

a = lector.nextInt();

double b = 0;

double c = 0;

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

int N, C;

N = lector.nextInt();

C = lector.nextInt();

b = b + (N \* C);

c = c + C;

}

c = c \* 100;

double x = b / c;

System.out.println(e.format(x));

break;

}

}

}

# Ejercicio 9

Iu-Di-Oh!

By Ricardo Oliveira, UFPR BR Brazil

Timelimit: 1

*Iu-di-oh!* is a card game really popular among kids! Every *Iu-di-oh!* player has his own deck containing many cards. Each card contains N attributes (such as power, speed, smartness, etc.). Attributes are numbered from 1 to N and are given as positive integers.

A match of *Iu-di-oh!* is always played by two players. At the beginning of the match, each player chooses exactly one card from his deck. Then, an attribute is randomly chosen. The player whose the chosen attribute is greater in the card he choose wins the match. If the such attribute is equal in both cards, there is a tie.

Marcos and Leonardo are in the big final of the Brazilian *Iu-di-oh!* championship. The great prize is a Dainavision (that is almost as good as a Plaisteition 2!). Given the deck of both players, the card each one chooses and the chosen attribute, determine the winner!

Input

The input contains several test cases. The first line of each test case contains an integer N (1 ≤ N ≤ 100), the number of attributes each card contains. The second line contains two integers M and L (1 ≤ M, L ≤ 100), the number of cards in Marcos’ and Leonardo’s deck, respectively.

Next M lines describe Marcos’ deck. His cards are numbered from 1 to M, and *i*-th line describes the *i*-th card. Each line contains N integers ai,1,ai,2,..., ai,N (1 ≤ ai,j ≤ 109). Integer ai,j indicates the *j*-th attribute of the *i*-th card.

Next L lines describe Leonardo’s deck. His cards are numbered from 1 to L and are described in the same way as Marcos’ deck.

Next line contains two integers CM and CL (1 ≤ CM ≤ M, 1 ≤ CL ≤ L), the cards chosen by Marcos and Leonardo, respectively. Finally, the last line contains an integer A (1 ≤ A ≤ N) indicating the chosen attribute.

The input ends with end-of-file (EOF).

Output

For each test case, print a line containing “Marcos” if Marcos wins the match, “Leonardo” if Leonardo wins the match, or “Empate” in the case of a tie (without quotes).

| Input Sample | Output Sample |
| --- | --- |
| 3 2 2 3 8 1 6 7 9 1 2 3 8 4 1 1 2 2 | Marcos |

package usb;

import java.io.BufferedReader;

import java.io.IOException;

import java.io.InputStreamReader;

/\*\*

\*

\* @author nico\_

\*/

public class Principal {

private static int convertir(String a) {

return Integer.parseInt(a);

}

private static String vencedor(int podermarcos, int poderleonardo) {

if (podermarcos > poderleonardo) {

return "Marcos";

}

if (podermarcos < poderleonardo) {

return "Leonardo";

}

return "Empate";

}

public static void main(String[] args) throws IOException {

System.out.println("Digite Los valores: ");

InputStreamReader inp = new InputStreamReader(System.in);

BufferedReader br = new BufferedReader(inp);

String numero;

String[] cartas;

String[] cartasEscogidas;

int[][] matrizmarcos;

int[][] matrizleonardo;

String[] carta2;

String[] carta1;

while ((numero = br.readLine()) != null) {

cartas = br.readLine().split(" ");

matrizmarcos = new int[Integer.parseInt(numero)][Integer.parseInt(cartas[0])];

matrizleonardo = new int[Integer.parseInt(numero)][Integer.parseInt(cartas[1])];

for (int i = 0; i < matrizmarcos[0].length; i++) {

carta1 = br.readLine().split(" ");

for (int j = 0; j < matrizmarcos.length; j++) {

matrizmarcos[j][i] = convertir(carta1[j]);

}

}

for (int i = 0; i < matrizleonardo[0].length; i++) {

carta2 = br.readLine().split(" ");

for (int j = 0; j < matrizleonardo.length; j++) {

matrizleonardo[j][i] = convertir(carta2[j]);

}

}

cartasEscogidas = br.readLine().split(" ");

int a = convertir(cartasEscogidas[0]) - 1;

int b = convertir(cartasEscogidas[1]) - 1;

String atributoesc = br.readLine();

int podermarcos = matrizmarcos[convertir(atributoesc) - 1][a];

int poderleonardo = matrizleonardo[convertir(atributoesc) - 1][b];

System.out.println(vencedor(podermarcos, poderleonardo));

break;

}

}

}

# Ejercicio 10

Dijkstra

By Abner Samuel P. Palmeira, IFSULDEMINAS BR Brazil

Timelimit: 1

In the game The Witcher, Sigismund Dijkstra is the leader of the Redanian Secret Service, because of this he is one of the most important people in the world.

In addition Dijkstra has a large treasure, which has several types of jewelry.

Dijkstra is very curious to know how many different types of jewelry his treasure has.

Knowing that you are the best programmer on the continent Dijkstra hired you to check how many different types of jewelry he has in his treasure.

Input

The entry consists of several lines and each contains a string describing one of Dijkstra's jewels. This string is composed only of the characters '(' and ')', the sum of the length of all the string does not exceed 106.

Output

Print how many different kinds of jewelry Dijkstra has.

| **Input Sample** | **Output Sample** |
| --- | --- |
| ((  ))  ((  ))  ( | 3 |

public static void main(String[] args) {

List<String> camino = new ArrayList<>();

Scanner lector = new Scanner(System.in);

String a;

boolean bandera = true;

do {

a = lector.next();

if (a.contains("(") || a.contains(")")) {

if (!camino.isEmpty()) {

for (int i = 0; i < camino.size(); i++) {

if (a.equals(camino.get(i))) {

bandera = true;

break;

}

}

} else {

camino.add(a);

}

if (!bandera) {

camino.add(a);

}

} else {

break;

}

bandera = false;

} while (!a.contains("(") || !a.contains(")"));

System.out.println(camino.size());

}

}

# Ejercicio 11

Web Browser

By Guilherme de Lima Bernardes, UNB BR Brazil

Timelimit: 1

Lucas is a pretty radical guy when it comes to software licenses. Since beginning his undergraduate degree in computer engineering, he seeks to develop all the tools he needs. All this started after bad experiences using proprietary software. Now he believes that a real programmer must be self-sufficient, that is, he must build all the programs he needs, from a simple calculator to his own operating system.

This semester, Lucas is studying the web systems development course. To continue his philosophy of life, using only software he built himself, Lucas is already programming his own web browser. Much of the work has been completed, but some functionality still needs to be finished.

Lucas' browser has a search field where the user can enter a keyword, and clicking a confirmation button it will redirect to another page with the results of his search. When some string is entered in the search field, Lucas wants his program to display, below, some options to auto complete this string according to the searches already performed by the user.

For example, if the words "algoritmos" and "algas" have already been searched, when typing the string "alg", the program should suggest the words "algorithms" and "algas". Therefore, for each string entered, the program should suggest previously searched words prefixed with this string. If any word is equal to the typed string, it should also be suggested.

Lucas is concerned about the amount of words his program can suggest, and the maximum size they can reach. So he asked you to help him by writing a program where given a few words already searched and a series of queries composed of a string, indicate how many words the browser should suggest to the user, and the length of the largest of these words.

Input

The input is composed of several test cases. The first line of a test case has an integer N (1 ≤ N ≤ 10^4) indicating the number of words that have already been searched by the Lucas’ program. Each of the next N lines contains a nonempty string of a maximum of 100 lowercase letters [a - z]. For each test case, N words are different. Then there will be an integer Q (1 ≤ Q ≤ 100) indicating the number of queries. Each of the next Q lines describes a query with a non-empty string of a maximum of 100 lowercase letters [a - z], representing a string entered in the search field.

Output

For each test case print Q lines, where the i-th row is the answer to the i-th query. The response of each query should be composed of two integers separated by space, representing, respectively, the number of words suggested by the program when entering the string indicated by the ith query, and the length of the largest word contained in that subset. If no words are suggested, print -1. Print a blank line at the end of each test case.

| **Input Sample** | **Output Sample** |
| --- | --- |
| 5  maratonaicpc  maraton  programacao  progress  inputs  3  marat  programacao  outputs | 2 12  1 11  -1 |

public class Principal {

public static void main(String[] args) {

Scanner lector = new Scanner(System.in);

int numero = lector.nextInt();

String[] a = new String[numero];

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

a[i] = lector.next();

}

lector.nextLine();

int cantidad\_b = lector.nextInt();

for (int j = 0; j < cantidad\_b; j++) {

String txt = lector.next();

int cantidad = 0;

int tamano = 0;

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

if (a[i].contains(txt)) {

cantidad++;

int aux = a[i].length();

if (aux > tamano) {

tamano = aux;

}

}

}

if (cantidad > 0) {

System.out.println(cantidad + " " + tamano);

} else {

System.out.println("-1");

}

}

}

}

# Ejercicio 12

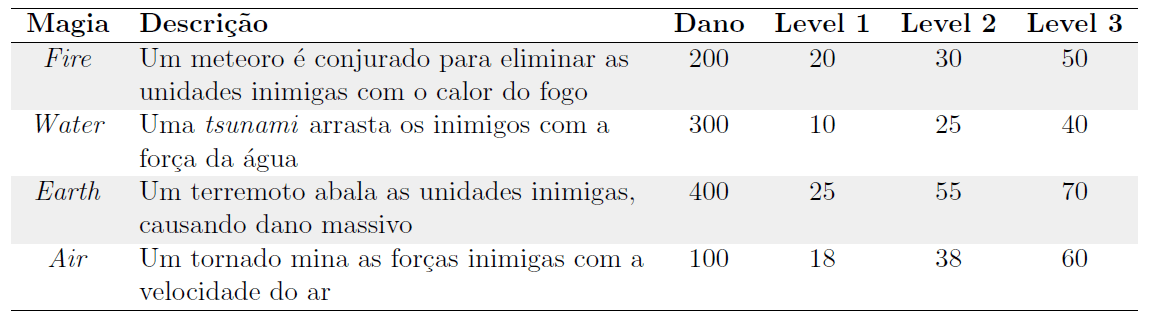
Magic and Sword

By Edson Alves da Costa Júnior, UNB BR Brazil

Timelimit: 1

In the Magic and Sword Tower defense, the player can cast area spells to defeat the enemy units. The spells are elemental: fire, water, air and earth, and the affected region is determined by a circle whose radius depends on the level of the spell.

The table below lists each spell, damage and its radius per level:



The enemy units are delimited by a rectangle of width **w** and height **h**, with the lower left corner positioned at the point (x0, y0). The enemy will suffer damage if their bounding rectangle has any intercession with the area defined by the spell circle.

Given the position and the bounding rectangle of the enemy unit, the center of the explosion, the identifier and level of the spell, determine the damage to the unit. If the unit is out of the spell range, the damage is equal to zero.

Input

The input consists of T (1 ≤ T ≤ 1000) test cases, where the value of T is reported in the first line of the input. Each test case consists of two lines. The first contains four integers representing the dimensions w and h (1 ≤ w, h ≤ 1000) of the rectangle and the coordinates x0 and y0 (0 ≤ x0, y0 ≤ 1000) from the lower left corner. The second line of the test case contains a string with the spell identifier (fire, water, earth and air), the level N of this spell (1 ≤ N ≤ 3) and the coordinates cx e cy (0 ≤ cx, cy ≤ 1000) from the center of the explosion area.

Output

For each test case, the output must be the value of the damage received by the unit, followed by a line break.

| **Input Sample** | **Output Sample** |
| --- | --- |
| 4  10 10 50 50  fire 1 85 55  10 10 50 50  fire 2 85 55  10 10 50 100  water 3 100 100  10 10 50 100  air 3 100 100 | 0  200  300  100 |

public class Principal {

public static void main(String[] args) {

Scanner lector = new Scanner(System.in);

Principal principal = new Principal();

System.out.println("Ingrese los valores w y h del rectangulo");

int w = lector.nextInt();

int h = lector.nextInt();

System.out.println("Ingrese las coordenadas X y Y");

int x0 = lector.nextInt();

int y0 = lector.nextInt();

System.out.println("Digite los valores cx y cy");

int cx = lector.nextInt();

int cy = lector.nextInt();

System.out.println("Digite el nivel");

int nivel = lector.nextInt();

System.out.println("Digite el hechizo");

String hechizo = lector.next();

principal.intermediario(w, h, x0, y0, cx, cy, nivel, hechizo);

}

public void fuego(int w, int h, int x0, int y0, int cx, int cy, int nivel, String hechizo) {

while (w <= 1000 & h <= 1000) {

while (x0 <= 1000 & y0 <= 1000) {

Integer dañoFuego = 200;

Integer dañoAgua = 300;

Integer dañoTierra = 400;

Integer dañoAire = 100;

if (hechizo.toUpperCase().equalsIgnoreCase("FIRE")) {

while (nivel <= 3) {

switch (nivel) {

case 1:

if (cx >= x0 & cy >= y0) {

System.out.println(dañoFuego = 0);

nivel = 0;

break;

} else {

System.out.println(dañoFuego = dañoFuego + 20);

break;

}

case 2:

if (cx >= x0 & cy >= y0) {

System.out.println(dañoFuego = dañoFuego + 0);

nivel = 0;

} else {

System.out.println(dañoFuego = dañoFuego + 30);

nivel = 0;

}

break;

case 3:

if (cx >= x0 & cy >= y0) {

System.out.println(dañoFuego = dañoFuego + 0);

nivel = 0;

} else {

System.out.println(dañoFuego = dañoFuego + 50);

nivel = 0;

}

break;

default:

break;

}

}

}

}

}

}

public void agua(int w, int h, int x0, int y0, int cx, int cy, int nivel, String hechizo) {

while (w <= 1000 & h <= 1000) {

while (x0 <= 1000 & y0 <= 1000) {

Integer dañoAgua = 300;

if (hechizo.toUpperCase().equalsIgnoreCase("WATER")) {

while (nivel <= 3) {

switch (nivel) {

case 1:

if (cx > x0 & cy > y0) {

System.out.println(dañoAgua = 0);

nivel = 0;

} else {

System.out.println(dañoAgua = dañoAgua + 10);

nivel = 0;

}

break;

case 2:

if (cx > x0 & cy > y0) {

System.out.println(dañoAgua = dañoAgua + 0);

nivel = 0;

} else {

System.out.println(dañoAgua = dañoAgua + 25);

nivel = 0;

}

break;

case 3:

if (cx >= x0 & cy >= y0) {

System.out.println(dañoAgua = dañoAgua + 0);

nivel = 0;

} else {

System.out.println(dañoAgua = dañoAgua + 40);

nivel = 0;

}

break;

default:

break;

}

}

}

}

}

}

public void tierra(int w, int h, int x0, int y0, int cx, int cy, int nivel, String hechizo) {

while (w <= 1000 & h <= 1000) {

while (x0 <= 1000 & y0 <= 1000) {

Integer dañoTierra = 400;

if (hechizo.toUpperCase().equalsIgnoreCase("EARTH")) {

while (nivel <= 3) {

switch (nivel) {

case 1:

if (cx > x0 & cy > y0) {

System.out.println(dañoTierra = 0);

nivel = 0;

} else {

System.out.println(dañoTierra = dañoTierra + 25);

nivel = 0;

}

break;

case 2:

if (cx > x0 & cy > y0) {

System.out.println(dañoTierra = dañoTierra + 0);

nivel = 0;

} else {

System.out.println(dañoTierra = dañoTierra + 55);

nivel = 0;

}

break;

case 3:

if (cx > x0 & cy > y0) {

System.out.println(dañoTierra = dañoTierra + 0);

nivel = 0;

} else {

System.out.println(dañoTierra = dañoTierra + 70);

nivel = 0;

}

break;

default:

break;

}

}

}

}

}

}

public void aire(int w, int h, int x0, int y0, int cx, int cy, int nivel, String hechizo) {

while (w <= 1000 & h <= 1000) {

while (x0 <= 1000 & y0 <= 1000) {

Integer damageAire = 100;

if (hechizo.toUpperCase().equalsIgnoreCase("AIR")) {

while (nivel <= 3) {

switch (nivel) {

case 1:

if (cx >= x0 & cy >= y0) {

System.out.println(damageAire = 0);

nivel = 0;

} else {

System.out.println(damageAire = damageAire + 18);

nivel = 0;

}

break;

case 2:

if (cx > x0 & cy > y0) {

System.out.println(damageAire = damageAire + 0);

nivel = 0;

} else {

System.out.println(damageAire = damageAire + 38);

nivel = 0;

}

break;

case 3:

if (cx >= x0 & cy >= y0) {

System.out.println(damageAire = damageAire + 0);

nivel = 0;

} else {

System.out.println(damageAire = damageAire + 60);

nivel = 0;

}

break;

default:

break;

}

}

}

}

}

}

private void intermediario(int w, int h, int x0, int y0, int cx, int cy, int nivel, String hechizo) {

if (hechizo.equalsIgnoreCase("Fire")) {

fuego(w, h, x0, y0, cx, cy, nivel, hechizo);

} else if (hechizo.equalsIgnoreCase("Water")) {

agua(w, h, x0, y0, cx, cy, nivel, hechizo);

} else if (hechizo.equalsIgnoreCase("Earth")) {

tierra(w, h, x0, y0, cx, cy, nivel, hechizo);

} else {

aire(w, h, x0, y0, cx, cy, nivel, hechizo);

}

}

}

Los datos se ingresan de esta forma:

Imagen que contiene captura de pantalla

Descripción generada automáticamente