Manejo de Datos

Datos de entrada

- En todos los problemas de las competencias los datos de entrada deben ser leídos de la entrada estándar.
- El formato en que vienen los datos es explicitado en el enunciado del problema.
- En los problemas recientes el formato de los datos ha sido estandarizado y estructurado por lo que es raro que se encuentren con problemas donde tengan que leer toda una línea para después parsearla.

Datos de salida

- En todos los problemas de las competencias los datos de salida deben ser impresos a la salida estándar.
- El formato en el que hay que imprimir los datos es explicitado en el enunciado del problema.
- Prestar mucha atención a los espacios y las líneas en blanco que pide el enunciado. No realizar esto correctamente resulta en *Presentation Error* lo que perjudica innecesariamente el puntaje.

Manejo de Datos en Python

Manejo de Datos en Python

Manejo de input en Python

- El manejo de input en python se hace mediante el método *input( )* el cual lee una línea y retorna un string.

```
number = int(input())
input1, input2 = map(int,input.split())
inputArray = map(int,input.split())
```

- En python2.7 tenemos el mismo método pero se llama *raw_input( ).*

Manejo de Datos en Python

Manejo de output en Python

- El manejo de output en python se hace mediante el método print() el cual puede recibir más de un parámetro e imprime los parámetros recibidos, separándolos con un espacio y agregando un salto de línea al final.

```
name = "Eric"
age = 74
print(f"Hello, {name}. You are {age}.")
'Hello, Eric. You are 74.'
```

- Tambien puede ser utilizado el print con formatted-string.
- La función *print( )* funciona como un wrapper del método stdout.write

Manejo de Datos en Python

Manejo mas eficiente

- Mejora de la eficiencia usando stdin.readline y stdout.write:

```
# input via readline method
n = stdin.readline()
# array input similar method
arr = [int(x) for x in stdin.readline().split()]
```

```
stdout.write(str(summation))
```

- write solo acepta string, por lo que debe convertirse en caso que se haga falta.

Link documentación

Manejo de Datos en C++

Manejo de Datos en C++

Existen dos formas de manejar la entrada y salida en este lenguaje:

- CIN - COUT
- SCANF - PRINTF

Manejo de Datos en C++

Manejo de entrada

- La función que nos provee C++ para leer la entrada es:

```
istream & operator >>
```

- Este operador trabaja sobre un stream de entrada. El stream asociado a la entrada estándar es std::cin.

- La ventaja de este operador es que lee el stream hasta encontrar el formato indicado por el segundo operando, por lo que usando el mismo operador podemos leer números, strings, caracteres, etc.

```
char c ;

string s;

int d;

std::cin >> s >> c >> d;
```

Manejo de Datos en C++

Uso de SCANF - PRINTF

- Uso de scanf/printf permite un tiempo más eficiente:

Código[GNU] C++	Sample Input	Sample Output

`#include<cstdio>`
`int TC, a, b;`
`scanf("%d", &TC); //nro de casos`
`while(TC --){ //repite hasta 0`
`  scanf("%d%d", &a, &b);`
`  //calcular la resp`
`  printf("%d \n", a + b);`
`}`
	3
	1 2
	5 7
	6 3
		3
		12
		9

Manejo de Datos en Java

Manejo de Datos en Java

Manejo de entrada

- Java nos provee una clase wrapper: java.util.Scanner
- Esta clase provee métodos para extraer distintos tipos de datos de un stream de datos, así como para saber si existen datos en el stream. En nuestro caso usaremos System.in como stream que es la entrada estándar.

```
Scanner in = new Scanner(new BufferedReader(new
                InputStreamReader
                (System.in)));
```

Link documentación

Manejo de Datos en Java

Manejo de salida

- El manejo de output en java se hace mediante el método *println*, *print* o *format* de la clase PrintStream. El objeto sobre el cual tendremos que llamar los métodos es *System.out*

- BufferedWritter envuelve el flujo de salida en un buffer, lo que significa que los datos se almacenan temporalmente en memoria antes de ser escritos en el flujo de salida, lo que mejora el rendimiento

```
BufferedWriter out = new BufferedWriter(new
OutputStreamWriter(System.out));

out.write("strings");

out.flush() // AL FINAL DEL PROGRAMA!
```

Link documentación