A - Easy Stack

Tienes una pila vacía y recibes algunas consultas. Estas consultas son las operaciones básicas como Push, Pop e imprimir el elemento Top. Ahora, debes procesar las consultas dadas.

Input

La primera línea contiene un número entero T ($0 \leq T \leq 10^6$).
Cada una de las siguientes T líneas contiene una consulta basada en estos formatos.

- 1 n: ingresa n ($0 < n \leq 10^9$) a la parte superior de la pila.
- 2: saca un elemento de la parte superior de la pila. Si la pila está vacía, no hagas nada.
- 3: imprime el elemento superior de la pila (ver formato de salida).

Output

Para cada consulta con formato 3, imprima el elemento superior de la pila. Si la pila está vacía, imprima "Empty!" sin comillas.

Ejemplo

Input

```
6
1 15
1 20
2
3
2
3
```

Output

```
15
Empty!
```

WARNING!: habrá una cantidad enorme de data para esto escriba las siguientes líneas dentro del main de la siguiente forma:

```
int main(){
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    cout.setf(ios::fixed);

    //Aqui va su código
}
```

B - Easy Queue

Te entregan una cola vacía y tu jefe te da algunas consultas. Estas consultas corresponden a las operaciones básicas de colas: Push, Pop e imprimir el elemento Front. Ahora, tu jefe te pide que proceses esas consultas.

Input

La primera línea contiene un entero T ($0 \leq T \leq 10^6$).
Cada una de las siguientes T líneas contiene una consulta con el siguiente formato:

- $1n$: ingresa n ($-10^9 \leq n \leq 10^9$) a la cola
- 2: saca el elemento del comienzo de la cola. Si la cola está vacía, no hagas nada.
- 3: imprime el elemento del comienzo de la cola (lea la sección Output)

Output

Por cada consulta del tipo 3, imprimir el valor del elemento del comienzo de la cola. Si la cola está vacía, imprima "Empty!" sin las comillas.

Ejemplo

Input

```
6
1 5
1 6
2
3
2
3
```

Output

```
6
Empty!
```

WARNING!: habrá una cantidad enorme de data para esto escriba las siguientes líneas dentro del main de la siguiente forma:

```
int main(){
    ios_base::sync_with_stdio(false);
    cin.tie(NULL);
    cout.setf(ios::fixed);

    //Aqui va su código
}
```

C - Backspace

Poco antes de que comenzara el concurso de programación, Bjarki decidió actualizar su computadora. No notó nada extraño hasta que comenzó a codificar en su editor favorito, Bim (Bjarki IMproved). Por lo general, cuando está escribiendo en un editor y presiona la tecla de retroceso, se borra un solo carácter a la izquierda. Pero después de la actualización, presionar esa tecla genera el carácter <. Probó todos los editores en su máquina, Bmacs, Neobim, bjedit, NoteBjad++ y Subjark Text, pero todos parecen tener el mismo problema. No tiene tiempo para buscar una solución en la web y, en cambio, decide eludir temporalmente el problema con un programa simple.

Ayude a Bjarki a escribir un programa que tome como entrada el string que se escribió en el editor de texto y genere el string como Bjarki pretendía escribirla. Puede suponer que Bjarki nunca tuvo la intención de escribir el carácter <, y que Bjarki nunca presionó la tecla de retroceso en una línea vacía.

Input

Una línea que contiene la cadena que se escribió en el editor de texto. La longitud de la cadena es como máximo 10^6 y solo contendrá letras minúsculas del alfabeto inglés, así como el carácter <.

Output

Imprime una sola línea que contiene el string como Bjarki pretendía escribirlo.

Ejemplos

Ejemplo 1

Input

```
a<bc<
```

Output

```
b
```

Ejemplo 2

Input

```
foss<<rritun
```

Output

```
forritun
```

Ejemplo 3

Input

```
a<a<a<aa<<
```

Output

NOTA: el ejemplo 3 efectivamente da un string vacío.

D - Balanceo de paréntesis

Se le da un string que consta de paréntesis () y []. Se dice que un string de este tipo es correcto:

- si este es un string vacío
- si el string A y el string B son correctos, entonces el string AB es correcto
  - Por ejemplo, A es () y B es [], entonces AB sería ()[] lo cual es correcto
- si el string A es correcto, (A) y [A] es correcto
  - Por ejemplo, A es (), entonces (A) sería (()) lo cual es correcto, y [A] sería [()] lo cual es también correcto.

Escriba un programa que reciba varios strings de este tipo y verifique que sean correctos. Tú programa puede asumir que el largo máximo del string es 128.

Input

La primera línea contiene un entero n.
Luego le siguen n líneas, donde cada una contiene un string que posée (,), [y].

Output

Por cada string debe escribir una línea que diga 'Yes', si el string esta correcto, y 'No' si el sting esta incorrecto.

Ejemplos

Ejemplo 1

Input

```
3
([])
(([()]))
([()[]()])()
```

Output

```
Yes
No
Yes
```