Task: SQU
Square™

english

CPSPC 2016, day 4. Available memory: 256 MB. *02.07.2016*

Mangolio became interested in making web pages. He already learned some HTML, CSS and even some JavaScript. After a while, he realized that the whole web thing is really complicated. Mainly HTML. It offers an incredible amount of ways to lay things out on a web page. All those *flows*, *blocks*, *flexes* and many more. He decided to create something simpler.

And so he did. He made a system that he called *Square*^{TM}, which is much simpler. It receives an expression (string) consising of symbols `.>v()` and based on that it creates a layout.

The basic building block of the system is a square. Mangolio decided to represent a square with `.` (a dot). More complicated shapes and layouts can be composed from squares. For this purpose, he came up with two operations: `>` – attach from right and `v` – attach from bottom.

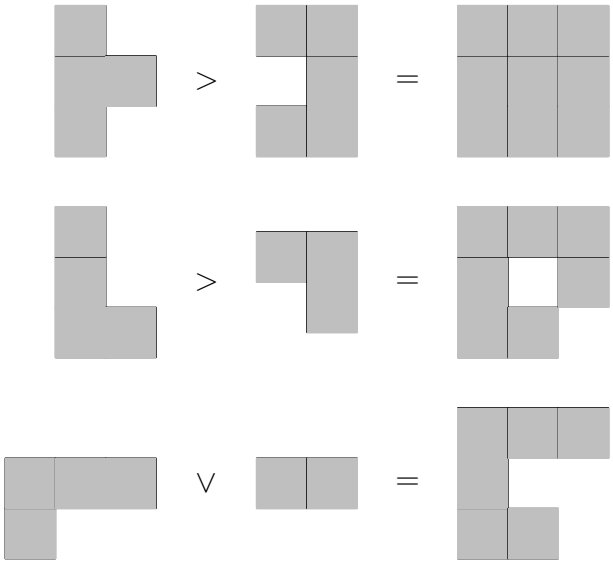
Operation *attach from right* works as follows: we take two shapes already composed by other operations, align them so that the top-most squares of both of them are in the same height and then we push them together until they are touching their sides.

Operation *attach from bottom* works similarly. We take two objects, align them according to their left-most squares and them push them together until they are touching.

Both of these operations are left-associative, meaning that the expression `.>.v.` is equivalent with `(.>.)v.`, but not with `.>(.v.)`.

Any expression in the *Square*^{TM} system can be put in parentheses `()` to become a single object. This object can then be combined with other objects.

The following pictures illustrate the basic usage of the *Square*^{TM} system.



Mangolio is very ambitious, so he started up a startup to appropriately monetize this new system. You are a young student searching for a small job. One day you stumbled upon an advertisement offering a job in a company called *Square*^{TM}*inc.*. Of course, you couldn't decline an offer from a company with such a marvelous name so you accepted it. Now you have to code the system.

You get an input string from the *Square*^{TM} system. Your task is to determine the position of each single square in the expression. Every square is 1×1 units large. The x axis increases to the right, while the y axis increases down. The left-most square should have its x coordinate equal to 0 and the top-most square should have its y coordinate equal to 0 as well.

Input

The input consists of one line which contains a single string s consisting only of symbols `.>v()`. This string is a valid expression for the *Square*^{TM} system. It holds that $|s| \leq 300\,000$.

Output

For each dot (square) in the input, output a single line. The i-th line should consist of two integers x_i and y_i – the position of the i-th square in the input expression. If j-th dot represents the left-most square, then $x_j = 0$. Also, if j-th dot represents the top-most square, then $y_j = 0$.

Examples

For the input data:

```
.>.
```

a correct result is:

```
0 0
1 0
```

For the input data:

```
.>.v.>.>.v.
```

a correct result is:

```
0 0
1 0
0 1
2 0
3 0
0 2
```

For the input data:

```
(.>(.v.)>.)>(.v(.>.)v.)v(.v(.>.)>(.v.))
```

a correct result is:

```
0 0
1 0
1 1
2 0
3 0
3 1
4 1
3 2
0 1
0 2
1 2
2 1
2 2
```

Grading

Subtask	Conditions	Points
1	$\|s\| \leq 3000$	10
2	$\|s\| \leq 10000$	20
3	$\|s\| \leq 50000$	20
4	no special restrictions	50

Úloha: SQU
Square™

czech

CPSPC 2016, Den 4. Dostupná paměť: 256 MB. *02.07.2016*

Mangólio se začal učit dělat webové stránky. Už umí psát HMTL, CSS a dokonce i JavaScript. Časem však zjistil, že je to celé velmi složité. Zejména HTML. Poskytuje nesčetné množství možností, jak rozložit objekty na stránce. Všechny ty *flow*-y, *block*-y, *flex*-y a co já vím, co ještě... Rozhodl se tedy vymyslet něco jednoduššího.

A skutečně vymyslel! Vytvořil systém *Square*^{TM}, který funguje velice jednoduše. Dostane výraz složený ze znaků `.>v()` a na základě něho vygeneruje rozložení stránky.

Základním stavebním kamenem jeho nového systému je čtverec. Čtverec se rozhodl reprezentovat znakem `.` (tečka). Ze čtverců lze vyskládat složitější útvary. Na tento účel si vymyslel dvě binární operace: `>` (přilož zprava) a `v` (přilož zespodu).

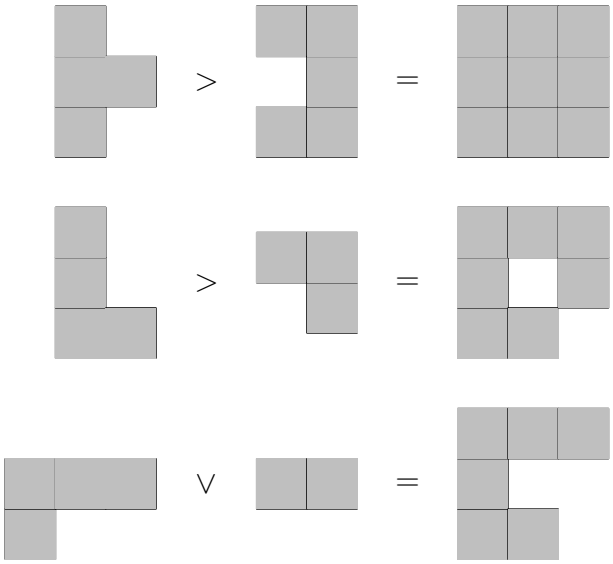
Operace *přilož zprava* vezme svoje argumenty, zarovná je tak, aby nejvyšší čtverec každého byl ve stejné výšce, a přiloží je k sobě tak, aby se dotýkaly jednou nebo více hranami.

Operace *přilož zespodu* funguje analogicky. Vezme svoje argumenty, zarovná je podle nejlevějších čtverců a přiloží je k sobě tak, aby se dotýkaly jednou nebo více hranami.

Obě tyto operace asociují zleva, tj. výraz `.>.v.` je ekvivalentní výrazu `(.>.)v.`, ale nikoli výrazu `.>(.v.)`.

Libovolný výraz v systému *Square*^{TM} je možné umístit do závorek `()` a tento složitější útvar dále spojovat s jinými útvary.

Následující obrázky ilustrují základní použití systému *Square*^{TM} na vytvoření rozložení.



Mangólio si založil startup, aby mohl svůj revoluční systém patřičně zpeněžit. Vy, mladý student hledající si brigádu, jste náhodou zakopl o inzerát nabízející práci ve firmě *Square*^{TM}*inc.*, a jelikož nabídka firmy s tak dobrým názvem se neodmítá, bez váhání jste ji přijali (aniž byste věděl, do čeho jdete). Teď musíte naprogramovat systém *Square*^{TM}.

Vaším úkolem je zjistit pozici každého jednotlivého čtverce. Každý čtverec má rozměry 1×1. x-ová osa je orientovaná směrem doprava a y-ová osa je orientovaná směrem dolů. Čtverec, který je nejvíc nalevo, nechť má x-ovou souřadnici 0; podobně čtverec, který je nejvýš, nechť má y-ovou souřadnici 0.

Vstup

Vstup obsahuje jeden řetězec s, složený ze znaků `.>v()`. Tento řetězec je platným výrazem systému *Square*^{TM}. Platí $|s| \leq 300\,000$.

Výstup

Vypište tolik řádků, kolik je teček (čtverců) na vstupu. Na i-tý řádek vypište výslednou x-ovou a y-ovou souřadnici i-té tečky ze vstupu. Připomínáme, že nejlevější čtverec má x-ovou souřadnici nulovou a nejvyšší čtverec má y-ovou souřadnici nulovou.

Příklad

Pro vstupní data:	je správný výstup:
`.>.`	`0 0` `1 0`
Pro vstupní data:	je správný výstup:

`.>.v.>.>.v.`	`0 0`
	`1 0`
	`0 1`
	`2 0`
	`3 0`
	`0 2`

Pro vstupní data:	je správný výstup:

`(.>(.v.)>.)>(.v(.>.)v.)v(.v(.>.)>(.v.))`	`0 0`
	`1 0`
	`1 1`
	`2 0`
	`3 0`
	`3 1`
	`4 1`
	`3 2`
	`0 1`
	`0 2`
	`1 2`
	`2 1`
	`2 2`

Hodnocení

Podúloha	Další omezení	Body
1	$\|s\| \leq 3000$	10
2	$\|s\| \leq 10000$	20
3	$\|s\| \leq 50000$	20
4	žádné speciální podmínky	50

Zadanie: SQU
Kwadrat™

polish

CPSPC 2016, dzień 4. Dostępna pamięć: 256 MB. *02.07.2016*

Bajtazar postanowił zająć się tworzeniem stron internetowych. Nauczył się HTML, CSS, a nawet trochę JavaScriptu. Szybko zorientował się jednak, że wszystko to jest zbyt skomplikowane – a szczególnie HTML. Sposobów ustawienia obiektów na stronie (kontenery, nakładki, bloki...) jest po prostu zbyt wiele. Bajtazar poczuł, że czas stworzyć coś łatwiejszego.

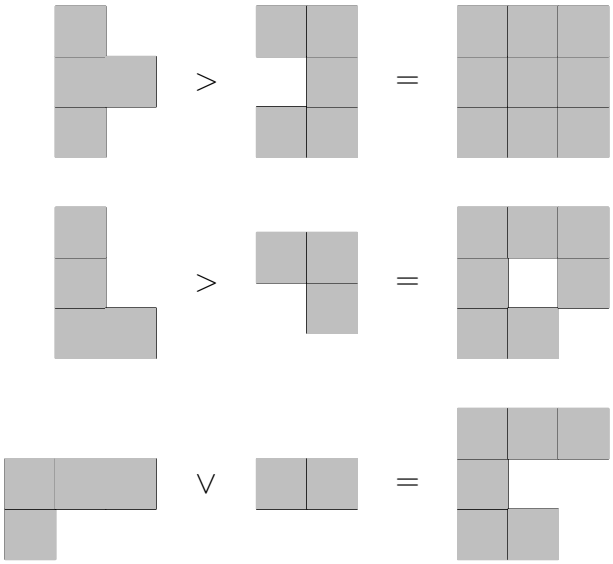
Nowy system Bajtazara nazywa się *Kwadrat*^{TM}, i jest niezwykle prosty. System otrzymuje wyrażenie (ciąg znaków) złożone z symboli `.>v()`, i na jego podstawie tworzy układ (*layout*) strony.

Podstawowym elementem systemu jest pojedynczy kwadrat, reprezentowany przez kropkę (`.`). Z kwadratów tworzy się bardziej skomplikowane układy, za pomocą dwóch operacji: łączenia z prawej (`>`) i łączenia z dołu (`v`).

Łączenie z prawej wygląda następująco: mając dwa kształty `A` i `B`, kształt `A>B` tworzymy ustawiając `A` i `B` tak, aby ich najwyższe kwadraty były ułożone na tej samej wysokości, po czym dosuwamy je do siebie najbliżej, jak to jest możliwe.

Łączenie z dołu działa podobnie. Mając dwa obiekty, ustawiamy je tak, aby ich skrajnie lewe kwadraty były w jednej linii, po czym zbliżamy je do siebie tak, najbardziej, jak się da.

Obrazki poniżej przedstawiają przypadki użycia systemu *Kwadrat*^{TM}.



Możemy w naszym ciągu zaznaczać kolejność operacji poprzez nawiasy, na przykład `.v(.>.)`. Obie podstawowe operacje są lewostronnie łączne (na przykład `.>.v.` oznacza to samo, co with `(.>.)v.`, ale nie to samo co `.>(.v.)`).

Dla danego ciągu zapisanego w systemie *Kwadrat*^{TM} oblicz, gdzie znajdzie się każdy kwadrat w ostatecznym układzie strony. Przypominamy, że kwadraty są w podanym wyrażeniu reprezentowane przez kropki, zakładamy że każdy kwadrat ma rozmiar 1×1. Przyjmij, że w ostatecznej figurze skrajnie lewe kwadraty mają współrzędną x równą 0, zaś najwyższe kwadraty współrzędną y równą 0 – współrzędne wszystkich innych kwadratów określ względem nich.

Wejście

Na wejściu znajduje się jeden wiersz, zawierający ciąg, w którym występują tylko znaki `.>v()`. Ciąg ten jest poprawnym wyrażeniem systemu *Kwadrat*^{TM}, a jego długość nie przekracza 300 000.

Wyjście

Dla każdej (reprezentującej kwadrat) kropki w łańcuchu wejściowym, wypisz wiersz zawierający dwie liczby całkowite x_i, y_i – współrzędne i-tego kwadratu w ostatecznie powstałym układzie. Kolejność wierszy powinna odpowiadać kolejności kropek na wejściu.

Przykłady

Dla danych wejściowych:

```
.>.
```

poprawnym wynikiem jest:

```
0 0
1 0
```

Dla danych wejściowych:

```
.>.v.>.>.v.
```

poprawnym wynikiem jest:

```
0 0
1 0
0 1
2 0
3 0
0 2
```

Dla danych wejściowych:

```
(.>(.v.)>.)>(.v(.>.)v.)v(.v(.>.)>(.v.))
```

poprawnym wynikiem jest:

```
0 0
1 0
1 1
2 0
3 0
3 1
4 1
3 2
0 1
0 2
1 2
2 1
2 2
```

Ocenianie

Podzadanie	Ograniczenia	Punkty
1	$\|s\| \leq 3000$	10
2	$\|s\| \leq 10000$	20
3	$\|s\| \leq 50000$	20
4	bez dodatkowych warunków	50

Úloha: SQU
Square™

slovak

CPSPC 2016, deň 4. Pamäťový limit: 256 MB. *02.07.2016*

Mangólio sa začal učiť robiť webové stránky. Už vie písať HMTL, CSS a dokonca aj JavaScript. Časom však zistil, že je to celé veľmi zložité. Najmä HTML. Ponúka nespočetné množstvo možností, ako rozložiť objekty na stránke. Všetky tie *flow*-y, *block*-y, *flex*-y a čo ja viem čo ešte. Rozhodol sa teda vymyslieť niečo jednoduchšie.

A veru aj vymyslel. Spravil systém *Square*^{TM}, ktorý funguje veľmi jednoducho. Dostane výraz zložený zo znakov `.>v()` a na základe neho vygeneruje rozloženie stránky.

Základným stavebným kameňom jeho nového systému je štvorec. Štvorec sa rozhodol reprezentovať znakom `.` (bodka). Zo štvorcov vie vyskladať zložitejšie útvary. Na tento účel si vymyslel dve operácie: `>` – prilož sprava a `v` – prilož zospodu.

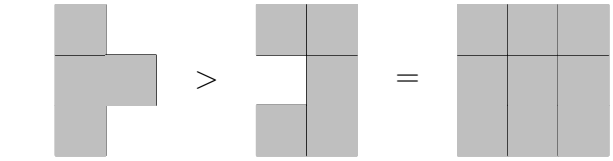
Operácia *prilož sprava* funguje tak, že zoberieme dva útvary, zarovnáme ich tak, aby najvyšší štvorec každého z nich bol v rovnakej výške a následne ich pritlačíme k sebe až kým sa nedotýkajú stenami.

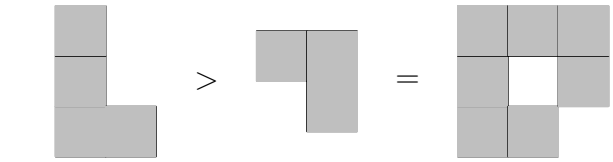
Operácia *prilož zospodu* funguje analogicky. Zoberieme dva útvary, zarovnáme ich podľa ich najľavejších štvorcov a potom ich pritlačíme k sebe až kým sa nedotýkajú.

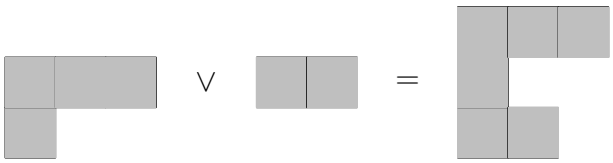
Obidve tieto operácie sú ľavo-asociatívne, čo znamená, že výraz `.>.v.` je ekvivalentný výrazu `(.>.)v.`, ale nie `.>(.v.)`.

Ľubovoľný výraz v systéme *Square*^{TM} je možné umiestniť do zátvoriek `()` a potom tento zložitejší útvar spájať s inými útvarmi.

Nasledujúce obrázky ilustrujú základné použitie systému *Square*^{TM} na vytvorenie rozloženia.







Mangólio si teda založil startup, aby mohol svoj nový systém patrične speňažniť. Vy, mladý študent hľadajúci si brigádu ste náhodou zakopli o inzerát ponúkajúci prácu vo firme *Square*^{TM}*inc.* a keďže ponuka firmy s tak dobrým názvom sa neodmieta, tak ste ju prijali (nevediac, do čoho idete). Teraz musíte naprogramovať systém *Square*^{TM}.

Vašou úlohou je zistiť pozíciu každého jednotlivého štvorca. Každý štvorec má rozmery 1×1. x-ová os je orientovaná smerom doprava a y-ová os je orientovaná smerom dole. Štvorec, ktorý je naľavejšie nech má x-ovú súradnicu 0 a štvorec, ktorý je najvyššie nech má y-ovú súradnicu 0.

Vstup

Na vstupe sa nachádza jeden reťazec s zložený zo znakov `.>v()`. Tento reťazec je platným výrazom systému *Square*^{TM}. Platí $|s| \leq 300\,000$.

Výstup

Vypíšte toľko riadkov, koľko je bodiek (štvorcov) na vstupe. Na i-tom riadku nech sú dve čísla: x_i a y_i určujúce pozíciu štvorca, ktorý je reprezentovaný na vstupe i-tou bodkou v poradí. Ak j-ta bodka reprezentuje najľavejší štvorec, nech platí $x_j = 0$ a ak j-ta bodka reprezentuje najvyšší štvorec, nech platí $y_j = 0$.

Príklady

Pre vstup:

```
.>.
```

je správny výsledok:

```
0 0
1 0
```

Pre vstup:

```
.>.v.>.>.v.
```

je správny výsledok:

```
0 0
1 0
0 1
2 0
3 0
0 2
```

Pre vstup:

```
(.>(.v.)>.)>(.v(.>.)v.)v(.v(.>.)>(.v.))
```

je správny výsledok:

```
0 0
1 0
1 1
2 0
3 0
3 1
4 1
3 2
0 1
0 2
1 2
2 1
2 2
```

Hodnotenie

Podúloha	Ďalšie ohraničenia	Body
1	$\|s\| \leq 3000$	10
2	$\|s\| \leq 10000$	20
3	$\|s\| \leq 50000$	20
4	žiadne špeciálne obmedzenia	50