Expressions regulars P94354_ca

Aquí tractem amb expressions regulars simplificades, les quals codifiquen paraules. En el que segueix, sigui P una paraula no buida formada només amb lletres minúscules, sigui D un dígit entre 1 i 9, i sigui $S = E_1, \ldots, E_n$ una seqüència no buida d'expressions regulars. Una expressió regular pot ser:

- `D[P]` : codifica D còpies de P.
- `D[S]` : codifica D còpies de la concatenació de les codificacions de E_1, ..., E_n.

Diversos exemples:

- `1[hola]` codifica `hola`.
- `3[hi]` codifica `hihihi`.
- `1[1[hola]3[hi]]` codifica `holahihihi`.
- `2[1[hola]3[hi]]` codifica `holahihihiholahihihi`.

Com podeu veure als exemples d'entrada, aquesta definició recursiva permet que hi hagi tants nivells com es vulgui de `[ ... ]`.

Entrada

L'entrada consisteix en diverses expressions regulars.

Sortida

Escriviu la paraula codificada per cada expressió regular donada.

Observació

Si s i t són strings, i c és un caràcter, aquestes operacions són vàlides:

```
s += t; // afegeix una copia de t a la dreta d's
s += c; // afegeix una copia de c a la dreta d's
```

Pista

La solució esperada és recursiva. El `main()` només conté:

```
char c;
while (cin >> c) cout << expressio(c) << endl;
```

Exemple d'entrada 1

```
1[hola]
3[hi]
1[1[hola]3[hi]]
2[1[hola]3[hi]]
2[1[a]3[bc]]
2[2[1[x]2[y]]]
1[1[2[a]1[b]3[c]]2[3[1[d]2[e]]1[f]]]
9[y]
2[2[2[2[2[z]]]]]
```

Exemple de sortida 1

```
hola
hihihi
holahihihi
holahihihiholahihihi
abcbcbcabcbcbc
xyyxyyxyyxyy
aabcccdeedeedeefdeedeedeef
yyyyyyyyy
zzzzzzzzzzzzzzzzzzzzzzzzzzzzzzz
```

Informació del problema

Autoria: Maria Blesa

Generació: 2026-01-25T12:05:55.284Z