Zastępowanie trójznaków

Współczesność przyzwyczaiła nas do sytuacji, w której wszystkie komputery na całym świecie posługują się tą samą konwencją kodowania znaków graficznych. Można zaryzykować twierdzenie, że wszelkie używane w tej chwili systemy reprezentacji znaków wywodzą się ze standardu nazywanego ASCII i co najmniej zestaw znaków o kodach dziesiętnych od 32 do 127 jest wszędzie taki sam.

Taki stan rzeczy różni się diametralnie od sytuacji, która panowała na rynku sprzętu komputerowego jeszcze w połowie lat sześćdziesiątych ubiegłego wieku. W tamtych czasach prawie każdy producent komputerów używał własnego standardu kodowania, przy czym gdyby różnice sprowadzały się tylko do różnic w kodach, byłby to problem o niewielkiej wadze. Gorzej, że poszczególne systemy różniły się znacząco także zestawami znaków (czyli alfabetami), co oznaczało, że znaki używane w jednym standardzie nie występowały w drugim i odwrotnie.

Najlepszym przykładem tej sytuacji może być używany przez firmę IBM system o wielce skomplikowanej nazwie EBCDIC (ang. *Extended Binary Coded Decimal Interchange Code*). We wczesnych wersjach tego standardu (co może wydać się nam niewiarygodne) nie występowały znaki '[' i ']' (nawiasy kwadratowe) oraz '{' i '}' (nawiasy klamrowe).

Nietrudno sobie wyobrazić wywołane tym komplikacje, ale zainteresujemy się tylko jedną z nich, której sedno zawarte jest w następującym pytaniu: „*jak można programować w języku C, jeśli na klawiaturze naszego komputera nie ma w ogóle ani nawiasów klamrowych, ani kwadratowych?*”.

Otóż można. Specjalnie dla poradzenia sobie z tym problemem wprowadzono do definicji języka C przedziwną konstrukcję, nazywaną „*trigraph*”, co możemy spróbować przetłumaczyć jako „trójznak”. Istota pomysłu jest następująca: nieszczęśliwy programista, któremu przypadło w udziale pisanie programu w języku C na komputerze, który cierpi na niedostatek koniecznych znaków, będzie przy pisaniu programu używał specjalnych, zdefiniowanych w standardzie języka, trzyznakowych kombinacji, a preprocesor, w ramach swoich normalnych czynności, będzie je zamieniał na znaki potrzebne do poprawnej konstrukcji programu. Do dzisiaj każdy szanujący się kompilator języka C potrafi operować trójznakami i co więcej, nadal dostępne są narzędzia, które potrafią przerobić klasyczny program w języku C na program z trójznakami i odwrotnie.

Zdefiniowano dziewięć trójznaków: oto one i ich znaczenia:

trójznak	**równoważnik**
??=	#
??/	\
??'	^
??(	[
??)	]
??!	|
??<	{
??>	}

??- ~

A tak wyglądałby program „Hello world!” zapisany w C przy użyciu trójznaków:

```
??=include <stdio.h>

int main(int argc, char *argv ??(??))??<

puts(”Hello world!”);

return 0;

??>
```

Prawda, że ładny?

Polecenie: napisz program, który wczyta ze standardowego wejścia program w języku C, zapisany z użyciem trójznaków i wypisze na standardowe wyjście ten sam program, ale zapisany klasycznie.

Dane wejściowe: pewna (nieznana z góry) liczba linii tekstu (każda o nieznanej z góry długości), składającego się na program w języku C zawierający trójznaki

Dane wyjściowe: program w języku C po zamianie trójznaków na znaki klasyczne

Przykład:

Wejście:

```
??=include <stdio.h>

int main(int argc, char *argv??(??))

??<

puts(”Hello world!”);

return 0;

??>
```

Wyjście:

```
#include <stdio.h>

int main(int argc, char *argv[] )

{

puts(”Hello world!”);

return 0;

}
```