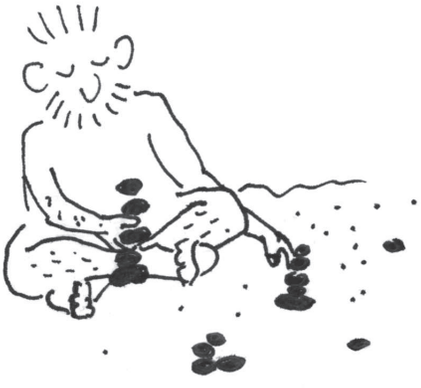
Najprostszy (?) komputer

Jak mało wystarczy mieć, żeby móc programować? Oto kurs *programów licznikowych* dla tych, którzy znają i tych, którzy nie znają się na programowaniu.

Program licznikowy to ciąg instrukcji. Operują one na zmiennych (licznikach), które mogą przyjmować wartości naturalne $(0, 1, 2, \ldots)$. Dopuszczalne instrukcje to:

$X{+}{+}$ – zwiększa wartość licznika X o 1
$X{-}{-}$ – zmniejsza wartość licznika X o 1 (nic nie robi, jeśli $X = 0$)
$X := 0$ – ustala X na 0



instrukcja sterująca:

jeśli $X = 0$ **skocz do** wybranego miejsca w programie

oraz, dla porządku, instrukcja **stop**, kończąca pracę programu.

Komputer wykonujący programy licznikowe można wyobrazić sobie jako zestaw stosów, na których leżą kamyczki (każdy stos to jeden licznik). Można opróżnić stos lub zdjąć albo dołożyć kamyk. Sterowanie pozwala skoczyć do innej instrukcji pod warunkiem, że testowany stos jest pusty.

Czy z tak ubogim arsenałem instrukcji jesteśmy w stanie cokolwiek zaprogramować? Spójrzmy na przykład na taki program:

$U := 0$
A: **jeśli** $Y = 0$ **skocz do** S
$Y{-}{-}$
$X{+}{+}$
jeśli $U = 0$ **skocz do** A
S: **stop**

Licznik U ma zawsze wartość 0 i jest potrzebny tylko po to, aby polecenie skoku do A było wykonywane zawsze (bezwarunkowo). Taką sztuczką z „głupim” licznikiem U będziemy się często posługiwać.

Powyższy program zdejmuje po jednym kamyku z Y i dokłada po jednym kamyku do X tak długo, dopóki są jeszcze kamyki w Y. Na końcu stos Y będzie pusty, można więc myśleć o tym programie jako o „przełożeniu stosu Y na X”. Żeby skrócić nieco następne programy, będziemy całą tę konstrukcję zapisywać w skrócie $X \leftarrow Y$.

Przy kopiowaniu wartość licznika Y „ginie”. Jeśli nie chcemy zgubić tej wartości, wystarczy, że posłużymy się dodatkowym licznikiem Z, który tymczasowo przechowa wartość Y, a na końcu ją odtworzy:

$U := 0$, $Z := 0$
A: **jeśli** $Y = 0$ **skocz do** S
$Y{-}{-}$
$X{+}{+}$
$Z{+}{+}$
jeśli $U = 0$ **skocz do** A
S: $Y \leftarrow Z$
stop

Przejdźmy do operacji arytmetycznych. Umiemy już dodać licznik Y do X (jak?). Odejmowanie jest równie proste: załóżmy, że chcemy policzyć $X - Y$ (gdzie $X > Y$):

$U := 0$
A: **jeśli** $Y = 0$ **skocz do** S
$Y - -$
$X - -$
jeśli $U = 0$ **skocz do** A
S: **stop**

Różnica będzie zapisana w X (oczywiście, gdybyśmy chcieli, moglibyśmy skopiować dane wejściowe X i Y do nowych liczników sposobem opisanym wcześniej i przeprowadzać działanie na kopiach, nie gubiąc X i Y podczas obliczeń). Mnożenie będzie odrobinę bardziej skomplikowane:

$U := 0$, $Z := 0$, $T := 0$
A: **jeśli** $X = 0$ **skocz do** S
$X - -$
B: **jeśli** $Y = 0$ **skocz do** C
$Y - -$
$Z + +$
$T + +$
jeśli $U = 0$ **skocz do** B
C: $Y \leftarrow T$
jeśli $U = 0$ **skocz do** A
S: **stop**

Wewnętrzna pętla (B) zwiększa Z o Y, równocześnie zapamiętując Y w T. Potem wartość Y jest odtwarzana. Całość jest powtarzana X razy. Wynik jest zapisany w liczniku Z.

Rozwiązania na stronie 24.

A teraz trochę zadań do samodzielnej pracy:

1. Napisz krótszy program, który mnoży X przez 2.

2. Napisz program obliczający $max(X, Y)$ dla danych liczników X i Y.

3. Napisz program, który dla danych dwóch liczników M i N oblicza iloraz I i resztę R z dzielenia M przez N. Postaraj się nie używać *żadnych* liczników poza M, N, I, R oraz „głupim” U.

4. Napisz krótszy program, który dzieli X przez 2...

5. ...i wykorzystaj swój pomysł do napisania programu, który sprawdza, czy X jest potęgą dwójki (odpowiedź – 1 jeśli tak, 0 jeśli nie – zapisz w specjalnym liczniku A).

Zadanie na jesienne wieczory. Wymyśl inne, pomysłowe (i krótkie) programy licznikowe realizujące różne operacje. A jeśli umiesz programować w innym języku, napisz interpreter programów licznikowych (czyli program, który wczytuje z pliku treść programu licznikowego i go wykonuje).

Programy licznikowe nie są może nadzwyczaj użyteczne ani szybkie, ale za to można się nimi dobrze bawić. Jednak nie o samą zabawę chodziło ich twórcom. Widzieliśmy już, że za pomocą liczników można zrealizować operacje arytmetyczne, przypisania, pętle, a więc całkiem sporo różnych konstrukcji programistycznych. Okazuje się, że każdy program obliczający pewien wynik na podstawie dostarczonych danych, który da się napisać w „poważnym” języku programowania (Pascalu, C czy Logo), można też zapisać w języku programów licznikowych. Mają one zatem taką samą siłę wyrazu jak prawdziwe komputery (temu stwierdzeniu można nadać bardzo ścisły, matematyczny sens). A na początku na to nie wyglądało, prawda?

Małą Deltę przygotował Michał ADAMASZEK