ZÁPADOČESKÁ UNIVERZITA V PLZNI

Fakulta aplikovaných věd

Katedra informatiky a výpočetní techniky

KIV/FJP

Cyklus DO – WHILE v jazyce PL/0
(implementace v PHP)

Autor: Antonín NEUMANN, A14N0139P

E-mail: neumann@students.zcu.cz

Datum odevzdání: 6. 1. 2015

Akademický rok: 2014/2015

1 Zadání

Zadání číslo 7 – cyklus *DO* příkaz *WHILE* podmínka.

Mezi klíčovými slovy *DO* a *WHILE* musí být právě jeden příkaz, má-li jich tam být více, musí být uzavřeny v bloku *begin-end*.

Implementace zadání ve skriptovacím jazyce PHP.

2 Úvodní analýza

V původní implementaci jazyka *PL/0*, jež má tato semestrální práce rozšířit, je k dispozici cyklus *WHILE-DO*. Z tohoto cyklu budu tedy vycházet, jediný rozdíl mezi cykly *DO-WHILE* a *WHILE-DO* je, že u cyklu *DO-WHILE* se příkazy provedou alespoň jednou. Podmínka se totiž ověřuje až na konci cyklu, vykonávání příkazů probíhá dokud je podmínka splněna.

2.1 Doplnění syntaxe jazyka PL/0

Původní syntaxi jazyka PL/0 je nutné rozšířit pouze o jediný řádek v bloku *příkaz* (vyznačeno tučně červenou barvou):

```
Příkaz ->  identifikátor ":=" výraz |
           call identifikátor |
           if podmínka then příkaz |
           while podmínka do příkaz |
           do příkaz while podmínka |
           begin příkaz { ";" příkaz } end |
           e
```

3 Implementace

Jelikož v jazyce *PL/0* existuje podmíněný skok pouze pokud je na vrcholu zásobníku 0, nabízela se implementace skoku podmínky při *WHILE* obráceně než by se na první pohled mohlo zdát. Pokud je podmínka uvedená ve *WHILE* nesplněná, potom dojde k přeskočení nepodmíněného skoku, který by způsobil skok na první příkaz v těle *DO*.

Zbytek je velmi analogický k již implementovanému cyklu *WHILE*. Je zde obrácená kontrola, která hlídá aby po *DO* následoval „ve správnou chvíli“ symbol *WHILE*. Dále, jelikož za *DO* následuje příkaz ukončený středníkem, bylo nutné implementaci o tuto kontrolu rovněž rozšířit.

3.1 Doplněný zdrojový kód

3.1.a Přidání chybové hlášky a přidání symbolu DO mezi povolené začáteční symboly

```
case 33:   echo 'ocekavano "while"';
           break;

$statbegsys[Kdosym] = 1;
```

3.1.b Zpracování podmínky

```
elseif($sym == Kdosym) {
    $cx1 = $cx;              /* adresa pocatku cyklu DO-W */
    getsym();                /* precteni nasledujiciho symbolu */
    $pom = nuluj();          /* vynulovani bin. promene $pom */
    sjednot($pom, $fsys);    /* preneseni priznaku z $fsys do $pom */
    $pom[Ksemicolon] = 1;    /* ukonceni po nalezeni stedniku (";") */
    statement($pom,$tx,$lev);   /* zavolani bloku statement */

    /* AN: nasledujici symbol musi byt stednik (";") */
    if ($sym == Ksemicolon) getsym(); else error(10);

    /* AN: nasledujici symbol musi byt "WHILE" */
    if ($sym == Kwhilesym) getsym(); else error(33);

    $pom = nuluj();          /* vynulovani bin. promene $pom */
    sjednot($pom, $fsys);    /* preneseni priznaku z $fsys do $pom */
    $pom[Ksemicolon] = 1;    /* ukonceni po nalezeni stedniku (";") */
    condition($pom,$tx,$lev);   /* zavolani bloku condition */
    $cx2 = $cx;              /* adresa instrukce podmineho skoku JPC */
    en(Kjpc,0,0);            /* generovani instrukce podmineneho skoku
                                (pokud je podminka 0) */
    gen(Kjmp,0,$cx1);        /* gen. instr. skoku na zacatek cyklu DO-W
                                (pokud je predchozi podminka 1) */
    $code[$cx2]->a = $cx;   /* nastaveni adresy podmineho skoku (JPC) na
                               adresu za nepopodmineny skok tedy na
                               instrukci nasledujici po cyklu DO-W */
}
```

3.2 Ukázkové programy v PL/0 a jejich překlad

3.2.a Příklad 1

```
var a;

begin
  a:=1;
  do a:=a+1; while a<3;
end.
```

```
 0  JMP     0     1
 1  INT     0     4
 2  LIT     0     1
 3  STO     0     3
 4  LOD     0     3
 5  LIT     0     1
 6  OPR     0     2
 7  STO     0     3
 8  LOD     0     3
 9  LIT     0     3
10  OPR     0    10
11  JMC     0    13
12  JMP     0     4
13  RET     0     0
```

3.2.b Příklad 2

```
var a,b;
begin
  a:=1;
  do begin
    a:=a+1;
    b:=2;
    do
      b:=b-1;
    while b>0;
  end;
  while a<3;
end.
```

```
  0  JMP         0         1
  1  INT         0         5
  2  LIT         0         1
  3  STO         0         3
  4  LOD         0         3
  5  LIT         0         1
  6  OPR         0         2
  7  STO         0         3
  8  LIT         0         2
  9  STO         0         4
 10  LOD         0         4
 11  LIT         0         1
 12  OPR         0         3
 13  STO         0         4
 14  LOD         0         4
 15  LIT         0         0
 16  OPR         0        12
 17  JMC         0        19
 18  JMP         0        10
 19  LOD         0         3
 20  LIT         0         3
 21  OPR         0        10
 22  JMC         0        24
 23  JMP         0         4
 24  RET         0         0
```

3.2.c Příklad 3

```
var a;

begin
  a:=1;
  do begin
    if (a%2)=1 then a:=a+2;
    if (a%2)=0 then a:=a+1;
  end;
  while a<30;
end.
```

```
  0  JMP         0         1
  1  INT         0         4
  2  LIT         0         1
  3  STO         0         3
  4  LOD         0         3
  5  LIT         0         2
  6  OPR         0         6
  7  LIT         0         1
  8  OPR         0         8
  9  JMC         0        14
 10  LOD         0         3
 11  LIT         0         2
 12  OPR         0         2
 13  STO         0         3
 14  LOD         0         3
 15  LIT         0         2
 16  OPR         0         6
 17  LIT         0         0
 18  OPR         0         8
 19  JMC         0        24
 20  LOD         0         3
 21  LIT         0         1
 22  OPR         0         2
 23  STO         0         3
 24  LOD         0         3
 25  LIT         0        30
 26  OPR         0        10
 27  JMC         0        29
 28  JMP         0         4
 29  RET         0         0
```

4 Závěr

Semestrální práce splňuje zadání a překladač si nyní poradí i s cyklem *DO-WHILE*. Pro vypracování úlohy mi velmi pomohl již implementovaný cyklus *WHILE-DO* a rovněž popis implementace překladače *PL/0* v PHP od Petra Sládka (dostupný na CW předmětu).