2003-11-26

9. Převod algoritmů do assembleru

9.1 Postup převodu algoritmu na program

- úprava (doplnění) algoritmu pro konkrétní programovací jazyk
- realizace základních struktur (sekvence, selekce, iterace) ve zvoleném jazyce
  - à zdrojový text v programovacím jazyku
- překlad do kódu počítače
  - à spustitelný program

9.1.1. Úprava algoritmu

V cvičném assembleru jsou k dispozici pouze rozhodování na základě porovnání střadače (S) s nulou, tj.

- S > 0 - instrukce: JP adresa
- S < 0 JN adresa
- S = 0 JZ adresa

Proto je nutná úprava podmínek typu:

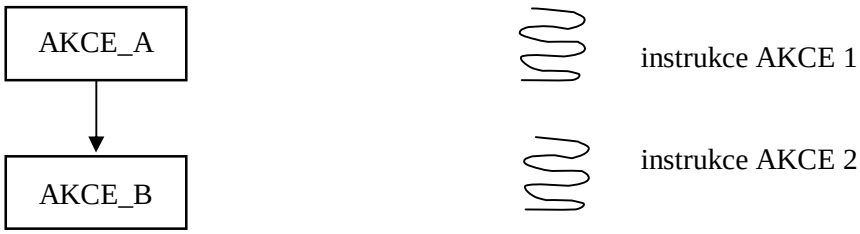
výraz > (<, =) hodnota

è **S := výraz - hodnota**
S > (<, =) 0

Př.:

I < 500	à	S := I – 500	…	S < 0
MOD > B	à	S := MOD – B	…	S > 0

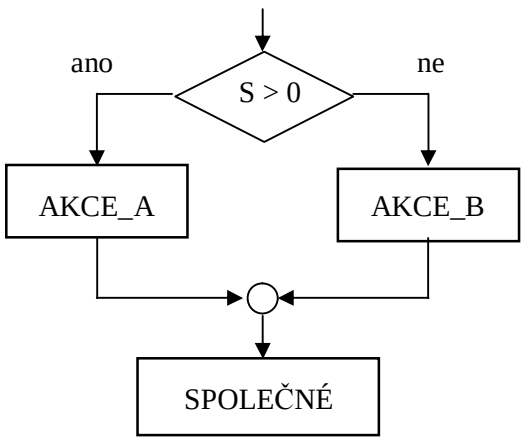
9.1.2. Realizace sekvence



Formát instrukce:

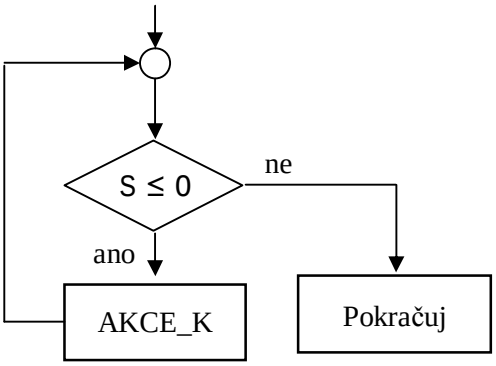
adresa	:	návěští	:	instrukce
200	:	start	:	LDA 100

9.1.3. Realizace selekce



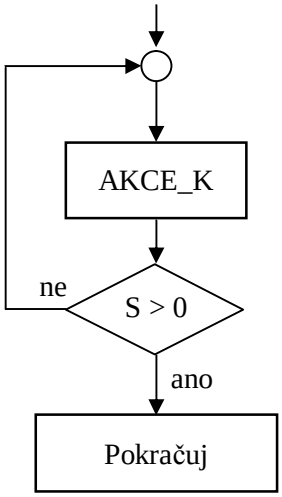

```
VYBER:   JP  AKCE_A
           instrukce
           AKCE_B
         JMP  SPOL
AKCE_A:    instrukce
           AKCE_A
SPOL:    společné
```

9.1.4. Realizace iterace s podmínkou na začátku




```
CYKL:    JP  POKRAC
           instrukce
           AKCE_K
         JMP  ZNOVA
POKRAC:  instrukce
         pokračuj
```

9.1.5. Realizace iterace s podmínkou na konci




```
CYKL:    instrukce
           AKCE_K
         JP  POKRAC
         JMP  ZNOVA
POKRAC:  instrukce
         pokračuj
```

6.2 Příklady

þ Př. 9.1:

Načtěte dvě hodnoty (čísla) A a B a vypište jejich součet S.

```
200:  START:   LDA  A
201:           ADD  B
202:           STA  SOUCET
203:  KONEC:   HLT

100:  A:
101:  B:
102:  SOUCET:
```

čti A
čti B
A + B **à** S
piš S

þ Př. 9.2:

Algoritmicky realizujte příkaz MOD A, B, kde MOD bude obsahovat zbytek po celočíselném dělení čísla A číslem B. A a B jsou celá kladná čísla.

```
200:  START:   LDA  A
201:           STA  MOD
202:  CYKL:    LDA  MOD
203:           SUB  B
204:           JN  KONEC
205:  TELO:    STA  MOD
206:           JMP  ZNOVA
207:  KONEC:   HLT

100:  A:
101:  B:
```

Čti A, B A **à** MOD
dokud MOD ≥ B (α)
MOD – B **à** MOD
piš MOD

```
102:    MOD:
```

þ Př. 9.3:

Algoritmicky realizujte příkaz DIV A, B, kde MOD bude obsahovat výsledek celočíselného dělení čísla A číslem B.

```
200:    START:      LNA  0
201:                STA  DIV
202:                LDA  A
203:                STA  MOD
204:    CYKL:       LDA  MOD
205:                SUB  B
206:                JN   KONEC
207:    TELO:       STA  MOD
208:                LNA  1
209:                ADD  DIV
210:                STA  DIV
211:                JMP  ZNOVA
212:    KONEC:      HLT

100:    A:          101:    B:          102:    MOD:        103:    DIV:
```

Čti A,B A **à** MOD
0 **à** DIV
dokud MOD ≥ B (α)
MOD – B **à** MOD DIV + 1 **à** DIV
piš DIV, MOD

þ Př. 9.4:

Upravte algoritmus z příkladu 9.3 tak, aby dával správný výsledek i pro záporné hodnoty čísel A a B.

```
200:    START:      LNA  0
201:                STA  DIV
202:    ABS_A:      LDA  A
203:                JN   ZAPOR_A
204:    KLAD_A:     LNA  1
205:                STA  ZM_A
206:                JMP  ABS_B
207:    ZAPOR_A:    LNA  -1
208:                STA  ZM_A
209:                MLP  A
210:                STA  A
211:    ABS_B:      LDA  B
212:                JN   ZAPOR_B
213:    KLAD_B:     LNA  1
214:                STA  ZM_B
215:                JMP  DIV_MOD
216:    ZAPOR_B:    LNA  -1
217:                STA  ZM_B
218:                MLP  B
219:                STA  B
220:    DIV_MOD:    LDA  A
221:                STA  MOD
222:    CYKL:       LDA  MOD
223:                SUB  B
224:                JN   KONEC
225:    TELO:       STA  MOD
226:                LNA  1
227:                ADD  DIV
228:                STA  DIV
229:                JMP  ZNOVA
230:    KONEC:      LDA  MOD
231:                MLP  ZM_A
232:                STA  MOD
233:                LDA  DIV
234:                MLP  ZM_A
```

Čti A, B
A < 0 (α) Ano: -A **à** A; -1 **à** zmA | Ne: 1 **à** zmA
B < 0 (β) Ano: -B **à** B; -1 **à** zmB | Ne: 1 **à** zmB
A **à** MOD
0 **à** DIV
dokud MOD ≥ B (γ)
MOD – B **à** MOD DIV + 1 **à** DIV
MOD * zmA **à** MOD
DIV * zmA * zmB **à** DIV
piš DIV, MOD

```
100: A:
101: B:
102: MOD:
```

```
235:                MLP  ZM_B       103: DIV:
236:                STA  DIV        104: ZM_A:
237:                HLT             105: ZM_B:
```

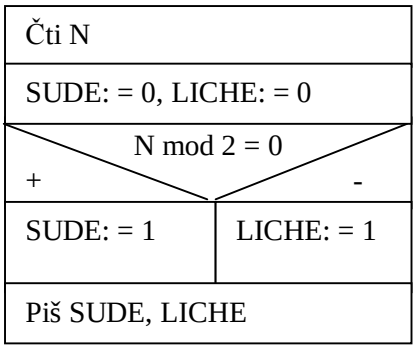
þ Př. 9.X:

Společné znění: Navrhněte a zakreslete pomocí plošných strukturogramů algoritmus, který…

þ Př. 9.5:

… rozhodne, zda načtené číslo N je sudé nebo liché.

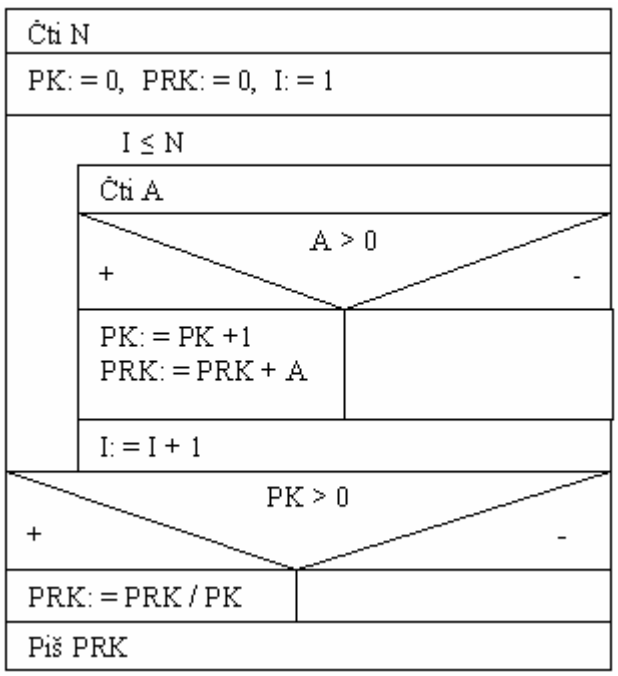
```
200:    START:      LNA  0
201:                STA  SUDE
202:                STA  LICHE
203:                LNA  2
204:                STA  DVA
205:                MOD  N, DVA
206:                JZ   SUD_C
207:    LICH_C:     LNA  1
208:                STA  LICHE
209:                JMP  KONEC
210:    SUD_C:      LNA  1
211:                STA  SUDE
212:    KONEC:      HLT

100:    N:
101:    SUDE:
102:    LICHE:
103:    DVA:
```



þ Př. 9.6:

… čte N vstupních čísel a vypočítává aritmetický průměr pouze z kladných čísel.

```
200:    START:      LNA  0
201:                STA  PK
202:                STA  PRK
203:                LNA  1
204:                STA  I
205:    CYKL:       LDA  I
206:                SUB  N
207:                JP PRUMER
208:    KROK:       LDA  A
209:                JP  A_KLAD
210:    A_ZAPOR:    JMP  K_POKR
211:    A_KLAD:     ADD  PRK
212:                STA  PRK
213:                LNA  1
214:                ADD  PK
215:                STA  PK
216:    K_POKR:     LNA 1
217:                ADD  I
218:                STI  I
219:                JMP  CYKL
220:    PRUMER:     LDA  PK
221:                JN  KONEC
222:                JZ  KONEC
223:    JK_KLAD:    LDA  PRK        101:    I:
224:                RDIV  PK        102:    N:
225:                STA  PRK        103:    PK:
226:    KONEC:      HLT             104:    PRK:
```

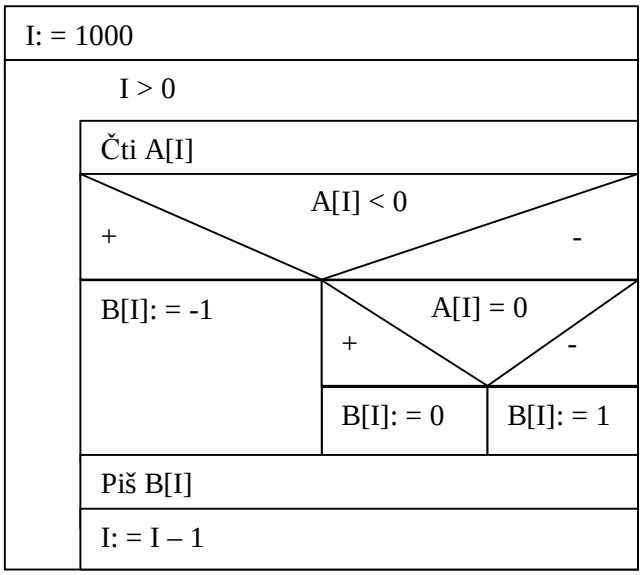


Poznámka: Instrukce pro „reálné“ dělení RDIV není v seznamu instrukcí cvičného strojového kódu, její význam by byl následující: <S> / <PK> **à** S

þ Př. 9.7:

… převede pole A, obsahující 1000 celých (kladných i záporných) čísel, na pole B tak, že všechna kladná čísla nahradí hodnotou +1, záporná hodnotou –1 a nuly hodnotou 0.

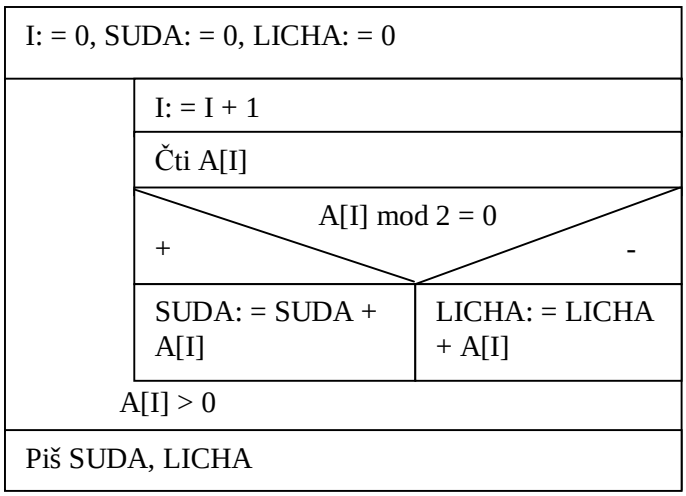
```
200:    START:      LNA  1000
201:                STI
202:    CYKL:       LDA  A[IND]
203:                JN   A_ZAPOR
204:                JZ   A_NULA
205:    A_KLAD:     LNA  1
206:                JMP  C_POKR
207:    A_ZAPOR:    LNA  -1
208:                JMP  C_POKR
209:    A_NULA:     LNA  0
210:    C_POKR:     STA  B[IND]
211:                LOOP CYKL
212:    KONEC:      HLT

1000:   A:
1001 – 2000:    pole A
3000:   B:
3001 – 4000:    pole B
```



þ Př. 9.8:

… v poli A, obsahujícím celá kladná čísla, kde není znám předem počet čísel v tomto poli – posledním prvkem pole je ale 0 (slouží jako zarážka), zjistí součet lichých a.součet sudých čísel.

```
200:    START:      LNA  0
201:                STA  SUDA
202:                STA  LICHA
203:                STI
204:                LNA  2
205:                STA  DVA
206:    CYKL:       LNA  1[IND]
207:                STI
208:                MOD  A[IND], DVA
209:                JZ   C_SUDA
210:    C_LICHA:    LDA  A[IND]
211:                ADD  LICHA
212:                STA  LICHA
213:                JMP  TEST
214:    C_SUDA:     LDA  A[IND]
215:                ADD  SUDA
216:                STA  SUDA
217:    TEST:       LDA  A[IND]
218:                JP   CYKL
219:    KONEC:      HLT

101:    SUDA:
102:    LICHA:
103:    DVA:
```

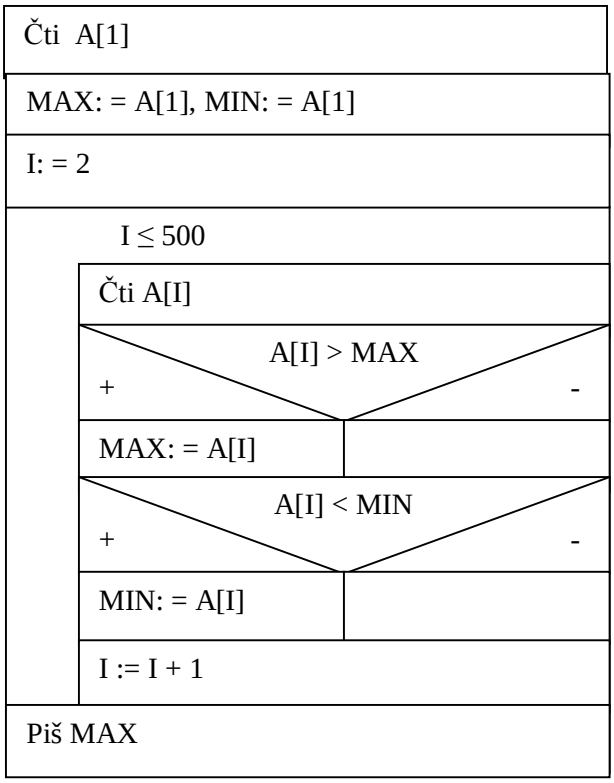


þ Př. 9.9:

… v poli A, obsahujícím 500 celých čísel (kladných i záporných), zjistí největší (maximum) a nejmenší (minimum) z těchto čísel.

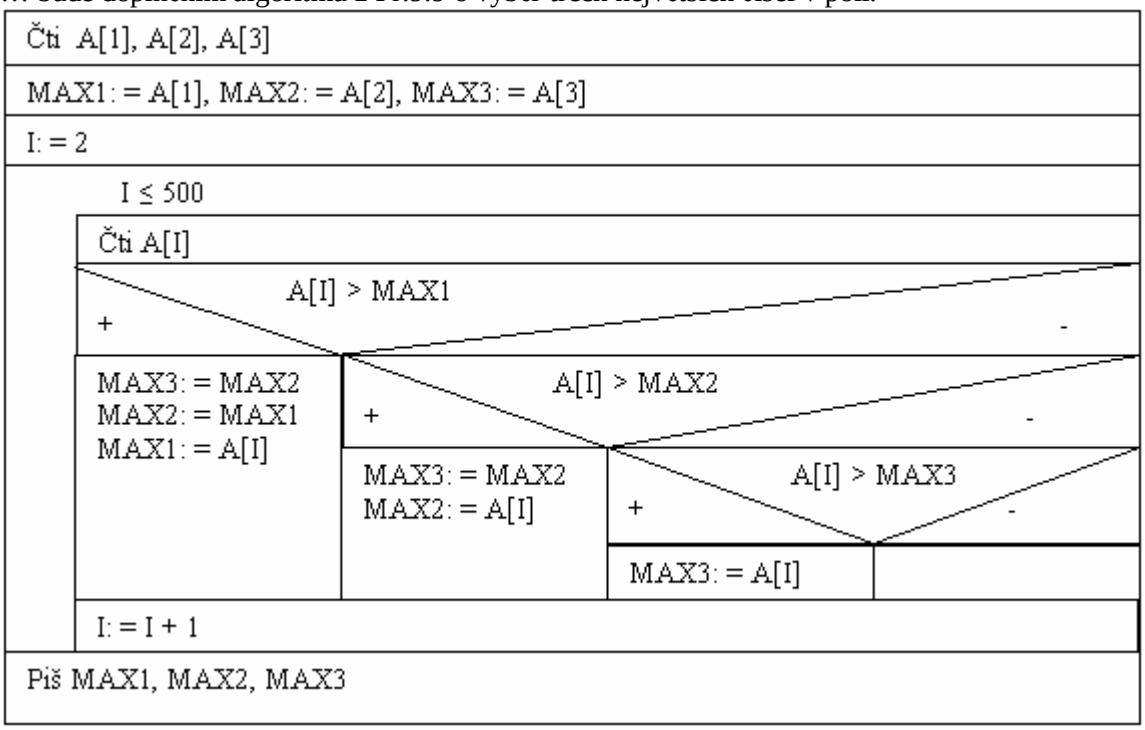
```
200:    START:      LNA  1
```

```
201:                STI
202:                LDA  A[IND]
203:                STA  MAX
204:                STA  MIN
205:                LNA  2
206:                STI
207:                LNA  500
208:                STA  N
209:    CYKL:       LNA  0[IND]
210:                SUB  N
211:                JP  KONEC
212:    VYB_MAX:    LDA  A[IND]
213:                SUB  MAX
214:                JP  JE_MAX
215:    NE_MAX:     JMP  VYB_MIN
216:    JE_MAX:     LDA  A[IND]
217:                STA  MAX
218:    VYB_MIN:    LDA  A[IND]
219:                SUB  MIN
220:                JN  JE_MIN
221:    NE_MIN:     JMP  POKRAC
222:    JE_MIN:     LDA  A[IND]
223:                STA  MIN
224:    POKRAC:     LNA  1[IND]
225:                STI
226:                JMP  CYKL
227:    KONEC:      HLT

100:    MAX:
101:    MIN:
102:    N:
```



Př. 9.10:

… bude doplněním algoritmu z Př.9.9 o výběr třech největších čísel v poli.




```
200:    START:      LNA  1
201:                STI
```

```
202:                LDA  A[IND]
203:                STA  MAX1
204:                LNA  2
205:                STI
206:                LDA  A[IND]
207:                STA  MAX2
208:                LNA  3
209:                STI
210:                LDA  A[IND]
211:                STA  MAX3
212:                LNA  2
213:                STI
214:                LNA  500
215:                STA  N
216:  CYKL: LNA  0[IND]
217:                SUB  N
218:                JP  KONEC
219:  VYB_MAX1:  LDA  A[IND]
220:                SUB  MAX1
221:                JP  JE_MAX1
222:  NE_MAX1:    JMP  VYB_MAX2
223:  JE_MAX1:    LDA  MAX2
224:                STA  MAX3
225:                LDA  MAX1
226:                STA  MAX2
227:                LDA  A[IND]
228:                STA  MAX1
229:  VYB_MAX2:  LDA  A[IND]
230:                SUB  MAX2
231:                JP  JE_MAX2
232:  NE_MAX2:    JMP  VYB_MAX3
233:  JE_MAX2:    LDA  MAX2
234:                STA  MAX3
235:                LDA  A[IND]
236:                STA  MAX2
237:  VYB_MAX3:  LDA  A[IND]
238:                SUB  MAX3
239:                JP  JE_MAX3
240:  NE_MAX3:    JMP  POKRAC
241:  JE_MAX3:    LDA  A[IND]
242:                STA  MAX3
243:  POKRAC:     LNA  1[IND]
244:                STI
245:                JMP  CYKL
246:  KONEC:       HLT

101:  MAX1:
102:  MAX2:
103:  MAX3:
104:  N:
```