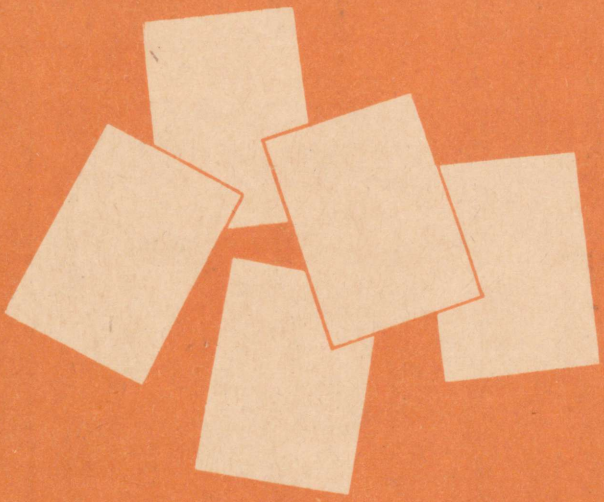

PROGRAMME KÕIGILE

II

TARTU
1969

TARTU RIIKLIK ÜLIKOOL

Arvutuskeskus

PROGRAMME KÕIGILE

II

ALGOL-mai

keele ja translaatori

kasutamisjuhend

Koostanud: K. Ääremaa

Ü. Kaasik

Tartu 1969

ПРОГРАММА ДЛЯ ВСЕХ

II

на эстонском языке

Составили Эзрема Куулдеп и Каазик Кыо

Тартуский государственный университет

ЭССР, г. Тарту, ул. Кликооли, I8

Vastutav toimetaja V. Tinn
Korrektor M. Raisma

==========

TRÜ rotaprint 1969. Paljundamisele antud 30. XII 1969. Trükipoognaid 3,5. Tingtrükipoognaid 3,26. Arvestuspoognaid 2,14. Trükiarv 750. Paber 30 x 42. 1/4. MB 08062. Tell. nr. 1003.

Hind 15 kop.

Algoritmilises keeles ALGOL kirjutatud programm määrab üheselt ülesande lahendusalgoritmi, kuid teda ei saa veel vahetult kasutada ülesande lahendamiseks elektronarvutil. Igal ALGOLi kasutaval arvutil peab olema spetsiaalne programm - translaator -, mis teisendab (transleerib) ALGOLis kirjutatud programmi arvuti käskude süsteemis esitatud programmiks. Transleerimise käigus ei täideta veel ühtegi programmi operaatorit ja translaatori töö lõpptulemusena saadakse ülesande lahendusprogramm arvuti käskude süsteemis. Ülesande lahendamiseks tuleb nüüd alles anda juhtimine koostatud programmi esimesele käsule.

Põhimõtteliselt on ALGOLis kirjutatud programm kasutatav igal elektronarvutil, millel on olemas vastava keele translaator. Erinevate arvutite jaoks koostatud translaatorid ei interpreteeri aga ühesuguselt ALGOLi (etaloonkeele) mõisteid ning seetõttu on tarvis teada, milliste kitsendustega (mõnikord ka täiendustega) kasutab konkreetne translaator etaloonkeelt, s.t. on tarvis teada vaadeldava translaatori sisendkeelt.

Järgnevas vaatlemegi üht ALGOLi baasil ehitatud sisendkeelt, mida nimetame ALGOL-mai keeleks ning mille jaoks on olemas arvutile "Ural-4" koostatud translaator.

Järgnevas käsitluses me ei tee vahet ALGOLi ja ALGOL-mai keele vahel siis, kui nad ühtivad. Seetõttu on sageli kasutatud lühemat nimetust ALGOL (pikema ALGOL-mai asemel).

I. SISENDKEELE KIRJELDUS

§ 1. ALGOL-MAI PÕHIELEMENDID.

Põhisümboliteks on ALGOL-mai keeles järgmised sümbolid:

1) suured ning väikesed ladina, kreeka ja vene tähed;
2) numbrid: 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 0;
3) sõnalised põhisümbolid: array | begin | do | end | else | for | go to | if | integer | out | real | switch | then | print | stop | while;
4) aritmeetiliste tehete märgid: + | − | × | / | ÷ | ↑;
5) võrdlusmärgid: = | ≠ | > | ≥ | < | ≤;
6) eraldavad märgid: [|] | (|) | . | ; | , | : | := | ₁₀ | ⌧ | ␣;

Sõnaliste põhisümbolite esiletõstmiseks tekstis kriipsutame nad alla kahe joonega.

§ 2. ARVUD.

Algoritmilises keeles ALGOL jagatakse arvud kahte põhitüüpi: täisarvud ja reaalarvud, kusjuures täisarvud jagunevad omakorda kaheks: märgita täisarvud ja märgiga täisarvud.

Märgita täisarv on numbrite jada, mille pikkus ALGOL-mai keeles ei tohi olla üle viie numbri. Märgita täisarvud on näiteks

12345

34

00210

7201

Rõhutame siinjuures, et näiteks arv 321476 on küll märgita täisarvuks ALGOLi etaloonkeeles, kuid ei ole seda ALGOL-mai keeles, sest selle arvu pikkus on suurem viiest numbrist.

Märgita täisarvu, mille ette on kirjutatud märk + või - nimetatakse märgiga täisarvuks. Märgiga täisarvudeks on näiteks

-73214

+27

-003

Reaalarv kujutab endast tavalist kümnendmurdu, milles täisosa on murdosast eraldatud punkti (mitte koma!) abil. Vaadeldavas keeles ei tohi täisosa ja murdosa pikkus kokku ületada kaheksat numbrit ning täisosa ja murdosa pikkus eraldi võetuna ei tohi ületada viit numbrit. Reaalarv võib alata ka kümnendpunktiga, kuid ei tohi sellega lõppeda. Reaalarvu ette võib kirjutada märgi + või -. Reaalarvud on näiteks

3.25

37283.274

+98.2

.37

-0.0

-0028.760

Reaalarve võib kirjutada ka nn. normaalkujul $M_{10}P$, kus M on mantiss ja P järk. Arvu mantissiks võib olla suvaline

reaalarv, järguks aga täisarv. Normaalkujul reaalarvudeks on näiteks järgmised:

$3.25_{10}+0$

$3728.3274_{10}1$

$0.982_{10}+2$

$-0.0_{10}0$

$-00287.60_{10}-1$

ALGOL-mai keeles võib normaalkujul arve kirjutada vahemikus ($0.92_{10}-18$, $0.92_{10}+19$). Vahemikus ($0.92_{10}-19$,$0.92_{10}-18$) asuvate arvude kasutamisel tuleb kirjutada mantiss 10 korda väiksemana ja vastavalt suurendada järku. Näiteks arvu $0.7_{10}-19$ asemel peab kirjutama $0.07_{10}-18$ (see nõue on tingitud kasutatava teisendusprogrammi iseärasustest).

Toome mõningaid näiteid <u>valesti</u> kirjutatud reaalarvudest:

34. - reaalarv ei tohi lõppeda kümnendpunktiga;

$34._{10}+1$ - mantiss ei tohi lõppeda kümnendpunktiga;

873241.3 - täisosa ei tohi sisaldada üle 5 numbri;

8.732413 - murdosa ei tohi sisaldada üle 5 numbri;

$34_{10}2.3$ - arvu järk peab olema täisarv;

34
+0034
-034 } - translaator vaatleb neid arve täisarvudena, mitte reaalarvudena.

Mingi arvu kirjutamine ALGOL-mai keeles ei ole muidugi üheselt määratud. Näiteks +200 ja 200 tähistavad sama arvu. Samuti ka 3.2, $0.32_{10}+1$ ja $0.32_{10}1$. Arvu kujuga on aga üheselt määratud tema tüüp. Igale täisarvule eraldab translaator programmis (20-kohalise) pesa, reaalarvule aga (40-kohalise) topeltpesa.

§ 3. MUUTUJAD.

Muutujate tähistamiseks kasutatakse ALGOLis identifikaatoreid. Identifikaatoriks võib olla suvaline tähtede ja numbrite jada, mis algab tingimata tähega. Identifikaatoriteks on näiteks

b3, СКОРОСТЬ, ß, DALLAS, valik1, valik2, B073;

Identifikaatoriteks ei saa olla aga näiteks järgmised kombinatsioonid:

3b1, 2, 2a - algavad numbriga!

a×b-1, A ↑ B, a(f), min␣arv - sisaldavad peale numbrite ja tähtede ka teisi märke!

Identifikaatoritena ei ole lubatud kasutada sõnalisi põhisümboleid, standardsete funktsioonide nimetusi (§ 4) ja sõna "ОТЛАД" (§ 14). Identifikaatori pikkus ei ole piiratud, kuid translaator kasutab nende eristamiseks ainult viit esimest sümbolit. Nii näiteks loetakse muutujad identifikaatoritega valik1 ja valik2 üheks ning samaks muutujaks.

Muutujad jagunevad ALGOL-mai keeles kahte klassi: lihtmuutujad ja indeksiga muutujad.

Lihtmuutuja on samastatud tema nimetusena kasutatava identifikaatoriga.

Indeksiga muutuja nimetus sisaldab peale identifikaatori veel ühe indeksi, mis on asetatud selle muutuja järele nurksulgudesse (indekssulgudesse). Erinevalt ALGOList võib ALGOL-mai keeles kasutada vaid ühe indeksiga muutujaid. Muutuja indeksiks võib olla ainult täisarv või täisarvude tüüpi lihtmuu-

tuja, kuid ei või olla täisarvuline muutuja märgiga + või –. Indeksiga muutujateks on näiteks

A[i], cd[3], b[-8].

Indeksiga muutujate hulka, millele on omistatud ühine identifikaator ja mis erinevad üksteisest ainult indeksi poolest, nimetatakse massiiviks, vastavat identifikaatorit aga massiivi identifikaatoriks.

Muutujate väärtusteks on arvud. Vastavalt sellele saavad muutujad üldse omandada vaid kahte tüüpi väärtusi. Üks ja sama muutuja saab (ühe bloki piires) omandada aga ainult ühte tüüpi väärtusi. Muutuja poolt omandatavate väärtuste tüüpi nimetatakse ühtlasi ka selle muutuja tüübiks. Muutuja tüüp määratakse kindlaks tüübikirjeldusega.

Lihtmuutuja tüübikirjeldus algab tüübi nimega, millele järgneb sellesse tüüpi kuuluvate lihtmuutujate loetelu. Tüüpide nimed on

integer (täisarvude tüüp),

real (reaalarvude tüüp).

Lihtmuutuja tüübikirjeldusteks on näiteks

integer a, g1, arv, summa;

real b, b1, abc;

Muutujad a, g1, arv ja summa võivad nende kirjelduste mõjupiirkonnas omandada ainult täisarvulisi väärtusi, muutujad b, b1, abc aga vaid reaalarvulisi väärtusi. Kui täisarvude tüüpi muutujale on programmis omistatud reaalarvuline väärtus, siis teisendab arvuti vastava arvu ümardamise teel täisarvuks (seejuures viiega lõppevad arvud ümardatakse ülespoole,

näiteks reaalarvulise väärtuse 3.5 omistamisel täisarvulisele muutujale g1 saadakse selle muutuja väärtuseks 4). Reaalarvude tüüpi muutujatele täisarvuliste väärtuste omistamisel teisendatakse täisarv reaalarvuks.

Arvuti liigse töö vältimiseks tuleb programmis jälgida, et muutujale omistatav väärtus oleks muutujaga sama tüüpi.

Indeksiga muutujate kirjeldamine toimub massiivide kaupa ning vastavat kirjeldust nimetatakse massiivikirjelduseks. Samasse massiivi kuuluvad indeksiga muutujad on seega alati ühte tüüpi. Massiivikirjeldus algab tüübi nimega ning sõnalise põhisümboliga array, millele järgneb vastavat tüüpi massiivide loetelu. Massiivide loetelu koosneb kirjeldatavate massiivide identifikaatoritest, mille järel indekssulgudes antakse indeksite muutumisrajad. Muutumisrajadeks võivad ALGOL-mai keeles olla vaid täisarvud, seejuures alumine raja ei tohi olla suurem kui ülemine. Kuna ALGOL-mai keeles kasutatakse ainult ühedimensionaalseid massiive, siis võib indekssulgudes olla vaid üks rajadepaar.

Sama tüüpi ning sama indeksite diapasooniga massiivide kirjeldamiseks võib kasutada ühtainust rajadepaari, kusjuures massiivide identifikaatorid eraldatakse üksteisest komadega ja ühised muutumisrajad tuuakse viimase massiivi järel. Sama rajadepaari abil kirjeldatavate massiivide arv ei tohi olla suurem kui 12. Näiteks massiivikirjeldused

integer array n[1:5]; integer array q[-3:2];

real array x[0:3],y[-20:0]; real array A,B[0:50];

reserveerivad mälus ruumi täisarvulistele muutujatele n_1, n_2, n_3, n_4, n_5, q_{-3}, q_{-2}, q_{-1}, q_0, q_1, q_2 ja reaalarvulistele

muutujatele x_0, x_1, x_2, x_3, y_{-20}, y_{-19}, ..., y_0, A_0, A_1,..., A_{50}, B_0, B_1, ..., B_{50}.

Muutuja kasutamiseks täisarvulisest massiivist tuleb massiivi identifikaatori järele indekssulgudesse kirjutada vastava elemendi indeks. Näiteks n[2] tähendab teist elementi (s.o. elementi n_2) eespool kirjeldatud massiivist, q[-1] aga vastava massiivi kolmandat elementi q_{-1}.

Reaalarvuliste massiivide korral tuleb indekssulgudesse kirjutada indeksi kahekordne väärtus. Näiteks kirjutades programmis x[6] tähendab see massiivi x elementi x_3.

Kahe ja enammõõtmeliste massiivide kasutamisel tuleb need massiivid kõigepealt taandada ühedimensionaalseteks, kodeerides nad selleks vastavalt ümber. Olgu näiteks programmis tarvis kasutada n-järku reaalsete elementidega ruutmaatriksit $A=(a_{ij})$. Kirjutades selle maatriksi elemendid näiteks veergude kaupa:

$$a_{11}, a_{21}, \ldots, a_{n1}, a_{12}, a_{22}, \ldots, a_{n2}, \ldots, a_{1n}, a_{2n}, \ldots, a_{nn},$$

saame n^2 elemendist koosneva ühedimensionaalse massiivi $a=(a_k)$, mille kirjelduse võib anda kujul

real array a [1:l];

kus l tuleb muidugi asendada n^2 arvulise väärtusega. Vaadeldava maatriksi mingi elemendi a_{ij} vasteks on nüüd indeksiga muutuja a[k], kus k=2(n(j-1)+i) (kui massiiv a oleks kirjeldatud täisarvulisena, siis saaksime k jaoks valemi k=n(j-1)+i). Ridade kaupa järjestatud elementidega maatriksite kasutamisel tuleb a_{ij} saamiseks võtta k=2(n(i-1)+j) (või täisarvulisel juhul k=n(i-1)+j).

§ 4. ARITMEETILISED AVALDISED.

Aritmeetilised avaldised jagunevad kahte liiki: aritmeetilised lihtavaldised ja tingimuslikud aritmeetilised avaldised. Käesolevas paragrahvis vaatleme vaid aritmeetilisi lihtavaldisi, lükates tingimuslike avaldiste käsitlemise edasi paragrahvini 8.

Aritmeetilise lihtavaldise defineerime induktiivselt:

1. Kui A on arv, muutuja või funktsioon, siis A on aritmeetiline lihtavaldis.
2. Kui A ja B on aritmeetilised lihtavaldised, kusjuures B ei alga märgiga + või - ning α on aritmeetilise tehte märk, siis A α B,+B ja -B on aritmeetilised lihtavaldised.
3. Kui A on aritmeetiline avaldis, siis (A) on aritmeetiline lihtavaldis.

Aritmeetilisteks avaldisteks on näiteks

a+b

$3.37_{10}+2$

200

-(c ↑ d)+summa1

a/b+(cxd) ↑ 2+3.37-(a ↑ 3)

+a

arcsin (x)

kuid aritmeetiliseks avaldiseks pole näiteks a ↑ -b, mille asemel tuleb kirjutada a↑(-b).

Funktsioonidena võib aritmeetilistes avaldistes kasutada vaid nn. standardfunktsioone, milleks ALGOL-mai keeles on

üksnes järgmised kümme:

sin(A), cos(A), sqrt(A), exp(A), ln(A), arcsin(A),
arctg(A), abs(A), sign(A), entier(A),

kus A tähendab aritmeetilist lihtavaldist. Funktsioonid sqrt(A) ja ln(A) on kasutatavad vaid siis, kui vastavalt $A \geq 0$ ja $A > 0$.

Funktsiooni argumendiks olev aritmeetiline avaldis tuleb tingimata kirjutada ümarsulgudesse. Näiteks kirjutist sinx ei tõlgenda translaator mitte standardfunktsioonina argumendiga x, vaid lihtsalt identifikaatorina.

Aritmeetiline avaldis kujutab endast mingi arvulise väärtuse leidmise eeskirja. Leitava väärtuse tüübi määramiseks tuleb selles aritmeetilises avaldises esinevad tehted nende sooritamise järjekorras läbi vaadates leida iga tehte tulemuse tüüp. Viimasena sooritamisele tuleva tehte tüüp ongi vaadeldava avaldise tüübiks.

Konstruktsioone, mille väärtustega sooritatakse mingit tehet, nimetatakse selle tehte operandideks. Tehte tulemuse tüübi määramiseks operandide tüüpide ja tehte märgi järgi tuleb kasutada järgmisi reegleid.

Tehete +, - ja × korral on tulemus täisarvude tüüpi vaid siis, kui mõlemad operandid on täisarvude tüüpi. Kui vähemalt üks operand on reaalarvude tüüpi, siis tulemus on samuti reaalarvude tüüpi.

Tehete / ja ↑ tulemus on alati reaalarvude tüüpi. Astendamine $a \uparrow b$ ei ole määratud juhtudel kui $a=0$, $b \leq 0$ või kui $a < 0$ ja b on reaalarvude tüüpi. ALGOL-mai keeles tuleb arvestada veel järgmist piiramist: kui astendajaks olev aritmeeti-

line avaldis sisaldab massiivi identifikaatorit, siis tuleb see avaldis võtta sulgudesse.

Täisarvuline jagamine on määratud ainult juhul, kui mõlemad operandid on täisarvude tüüpi. Jagatis a ÷ b leitakse valemi

$$a \div b = \mathrm{sign}(a/b) \times \mathrm{entier}(\mathrm{abs}(a/b))$$

abil.

Erinevalt ALGOLi etaloonkeelest ei ole ALGOL-mai keeles aritmeetilises avaldises tehete eelisjärjekorda ja kõik tehted sooritatakse nende esinemisjärjekorras vasemalt paremale. Tehete täitmise järjekorra muutmiseks kasutatakse ümarsulge. Näiteks kirjutist

$$a + b \times c + d \times 1 \uparrow m$$

tõlgendab translaator avaldisena

$$\left\{\left[(a + b) \cdot c + d\right] \cdot 1\right\}^{m} ,$$

kuigi etaloonkeeles see kirjutis tähendaks avaldist

$$a + bc + d1^{m}.$$

Viimase avaldise kirjutamiseks ALGOL-mai keeles tuleb kas kasutada sulge:

$$a + (b \times c) + (d \times (1 \uparrow m)) ,$$

või muuta liikmete järjekorda:

$$1 \uparrow m \times d + (b \times c) + a .$$

Arvuti liigse töö vältimiseks transleerimisel ja ülesande lahendamisel tuleb võimalikult vähendada sulgude arvu. Seega kuigi translaator tõlgendab äsja toodud kaht kirjutist

sama avaldisena, on viimane kirjutis siiski eelistatavam.

Aritmeetilise avaldise koostamisel tuleb veel arvestada, et programmis ei tohi kusagil olla enam kui 39 samaaegselt avatud sulgu (kaasa arvatud nii ümarsulud kui ka operaatorsulud).

§ 5. OPERAATORI MÕISTE, MÄRGEND JA TÜHIOPERAATOR.

Operaatoriks nimetatakse teatud terviklikku konstruktsiooniühendit, mis teostab mingeid algoritmis ettenähtud operatsioone. ALGOL-mai keeles on kokku 11 erinevat liiki operaatorit. Nendeks on: omistamisoperaator, tühioperaator, suunamisoperaator, tingimuslik operaator, tsüklioperaator, koodoperaator, peatusoperaator, operaator print, operaator out, liitoperaator ja blokk. Viimased kaks operaatoriliiki kujutavad sealjuures operaatorite jadasid, mis on operaatorsulgude begin ja end abil ühendatud üheks tervikuks. Erinevus liitoperaatori ja bloki vahel seisneb selles, et blokk peab peale operaatorite sisaldama ka kirjeldusi.

Operaatori lõpu tunnuseks on ALGOL-mai keeles kas semikoolon või else või end (sümbolile end peab sealjuures alati järgnema semikoolon, end või else, sest vastupidisel juhul vaadeldakse järgnevat programmilõiku kommentaarina). Operaatorite täitmine toimub üldiselt sellises järjekorras, nagu nad on kirjutatud programmis, kuid üheks algoritmides teostatavaks operatsiooniks võib olla ka operaatorite täitmise järjekorra muutmine.

Selleks et operaatorite täitmise loomuliku järjekorra

muutmine osutuks tehniliselt teostatavaks, tuleb mõningatele operaatoritele (millest jätkatakse operaatorite täitmist) omistada nimi. Niisugune nime andmine ehk märgendamine toimub sel teel, et operaatori ette kirjutatakse märgend koos temale järgneva kooloniga.

Märgendiks võib ALGOL-mai keeles olla suvaline tähtede ja numbrite kombinatsioon, mis erinevalt ALGOList võib alata ka numbriga. Märgendeid identifitseeritakse viie esimese sümboli järgi. Analoogiliselt identifikaatoritega ei tohi märgendina kasutada sõnalisi põhisümboleid ja standardfunktsioonide nimetusi (märgendina kasutatud sõnal "ОТЛАД" on eritähendus).

Märgendatud operaatoriteks on näiteks

A1: α;

13: β;

M3: begin α; β end ;

kus α ja β on suvalised operaatorid, A1, 13 ja M3 aga märgendid. Märgendada võib ka märgendatud operaatoreid, s.t. ühel operaatoril võib olla mitu erinevat märgendit, näiteks

M1 : M2 : A : α ;

Tuleb aga jälgida, et sama bloki erinevatel operaatoritel ei oleks ühesuguseid märgendeid, sest vastasel juhul ei oleks operaatorite täitmise järjekorra muutmine määratud üheselt.

Tühioperaator on selline operaator, mis ei sisalda ühtegi sümbolit ja ei teosta seega ka ühtegi operatsiooni.

§ 6. OMISTAMISOPERAATOR.

Omistamisoperaator on ette nähtud uute väärtuste andmiseks muutujatele. Selle operaatori üldkuju on

M := A;

kus M on muutuja ja A kas aritmeetiline avaldis või märgendamata omistamisoperaator. Omistamisoperaatori täitmisel leitakse kõigepealt selle operaatori koosseisu kuuluva aritmeetilise avaldise A väärtus ja omistatakse see sümbolist := vasakul paiknevale muutujale. Omistamisoperaatori definitsioonist tuleneb, et ühe ja sama aritmeetilise avaldise väärtuse omistamisel mitmele muutujale võime omistamisoperaatori vasakule poole kirjutada mitu üksteisest omistamismärgiga eraldatud identifikaatorit, kusjuures aga kõik need muutujad peavad olema sama tüüpi.

Näiteks operaator k := k + 1 × 2 ; omistab identifikaatoriga k tähistatud muutujale aritmeetilise avaldise k + 1 × 2 väärtuse (s.t. 2·(k+1)), operaatori x := y := z[4] := 1.0 ; toimel omandavad muutujad x, y ja z[4] väärtuse 1.0 ning peavad kõik olema reaalarvude tüüpi (seda tüüpi peab olema muidugi kogu massiiv z).

Igale programmis kirjeldatud reaalarvude tüüpi muutujale eraldab translaator ühe topeltpesa, täisarvude tüüpi muutujale aga lühipesa. Omistamisoperaatori täitmine on samaväärne omistatava väärtuse saatmisega mälu vastavasse pessa. Kui omistatava väärtuse tüüp ei ühti muutuja tüübiga, millele see väärtus omistatakse, siis vastava teisenduse teostab arvuti

automaatselt. Transleeritud programmi optimiseerimiseks tuleb aga jälgida vastavate tüüpide kokkulangemist, eriti konstantide omistamisel. Kui näiteks muutuja x on kirjeldatud reaalarvude tüüpi muutujana, siis kirjutis x := 1 ; on küll lubatav, kuid ebaotstarbekohane - õigem on kirjutada x:=1.0;.

Omistamisoperaatori paremal pool olevate identifikaatorite ja indeksite summa ei tohi olla suurem kui 24. Arvud 0 ja 0.0 ei tohi olla kirjutatud kahe järjestikuse konstandina, kui vähemalt ühte neist ei ole selles programmis juba kasutatud.

§ 7. SUUNAMISOPERAATOR.

Operaatorite täitmise järjekorra muutmiseks kasutatakse suunamisoperaatorit, mille üldkuju on

go to ℌ ;

kus ℌ tähendab nn. märgendiavaldist. Märgendiavaldiseks (ehk nimeliseks avaldiseks) nimetame avaldist, mille väärtuseks võivad olla vaid märgendid. Kõige lihtsamaks märgendiavaldiseks on märgend. Seega on suunamisoperaatoriteks näiteks operaatorid

go to M ;

go to KORD ;

kus M ja KORD on märgendid.

Suunamisoperaatori täitmise tulemusena antakse juhtimine operaatorile, mis on märgendatud vastava märgendiavaldise väärtuseks oleva märgendiga.

Märgendiavaldises võib märgend olla antud ka erilise in-

5.

deksiga muutuja - lüliti - abil. Selle muutuja indeksi väärtusteks võivad olla vaid naturaalarvud, muutuja väärtusteks aga märgendid. Konkreetne märgendi väärtus leitakse nn. lülitikirjelduse abil, mis peab olema antud vastava bloki alguses. Lülitikirjeldus algab sõnalise põhisümboliga switch, millele järgneb lüliti identifikaator (selleks võib olla suvaline identifikaator) ning pärast sümbolit := märgendite loetelu. Lülitikirjelduseks on näiteks:

switch q := M1, M2, S, Q ;

Lüliti indeksi väärtus näitab, mitmes märgend vastavas lülitikirjelduses on selle märgendiavaldise väärtuseks (kui indeksi väärtus on suurem märgendite arvust vastava lüliti kirjelduses, siis töötab transleeritud programm valesti). Näiteks eespool toodud lüliti q kirjeldust arvestades saame suunamisoperaatoris go to q[3] ; märgendiavaldise väärtuseks märgendi S ja täitmisele tuleb tegelikult operaator go to S; Analoogiliselt programmilõik

i := 4 ; go to q[i] ;

tähendab, et täidetakse operaator go to Q ;

Toome veel lüliti kasutamise selgitamiseks näite, leides polünoomi

$$P = az^2 + bz + c$$

järgmises kolmes punktis arvutatud väärtuste summa:

$$z_1 = \frac{a+b}{2} ; \quad z_2 = \frac{a-b}{2} ; \quad z_3 = \frac{a^3+b^3}{8} .$$

Polünoomi arvutamise eeskirja kolmekordse kirjutamise

vältimiseks kasutame lülitit, millega programmi vastav blokk omandab järgmise kuju:

```
begin real z,P;
      integer i;                          } kirjeldused
      switch q := M1,M2,M3;               }
z:=a+b/2; i:=1; go to R;                  z1 väärtuse leidmine
M1:S:=P; z:=a-b/2; i:=2;                  z2 väärtuse leidmine
      go to R;
M2:S:=S+P;z:=0.125x(a↑3+(b↑3));           z3 väärtuse leidmine
      i:=3; go to R;
R:P:=z↑2xa+(bxz)+c; go to q i ;           polünoomi väärtuste
M3:S:=S+P                                 arvutamine
end;                                      bloki lõpp
```

§ 8. TINGIMUSLIKUD OPERAATORID.

Võrdluseks nimetatakse loogilist avaldist, mis koosneb kahest võrdlusmärgiga ühendatud aritmeetilisest lihtavaldisest. Seega, kui A ja B on aritmeetilised lihtavaldised, siis võrdlusteks osutuvad

$$A > B,\quad A \geq B,\quad A < B,\quad A = B,\quad A \neq B \text{ ja } A \leq B.$$

Kui võrdluse paremaks pooleks olev aritmeetiline avaldis sisaldab märkidest + ja - erinevat tehtemärki või kui ta algab kas märgiga + või märgiga -, siis tuleb see avaldis ALGOL-mai keeles võtta sulgudesse. Seega näiteks võrdlused

$$a > +1$$

$$a - 2 \leqslant 2 \times i$$

on kirjutatud valesti. Õige kirjutusviis on

$$a > (+1)$$

$$a - 2 \leqslant (2 \times i).$$

Kahe reaalarvulise suuruse võrdlemiseks ei ole muide soovitatav kasutada märki =, sest võrreldavate arvude erinevus võib olla tingitud ümardamisest. Selle asemel on otstarbekohasem näiteks nõuda, et nende suuruste vahe absoluutväärtus oleks väiksem teatud küllalt väikesest arvust.

Tingimuslik operaator võimaldab vastavalt võrdluse tulemusele kas täita või jätta täitmata tema koosseisu kuuluvaid operaatoreid. Tingimusliku operaatori lihtsaima erijuhu moodustab nn. kui-operaator, mille üldkuju on

if α then A ;

kus α tähendab võrdlust ja A operaatorit (kuid mitte tsükli-operaatorit või tingimuslikku operaatorit). Kui-operaator töötab operaatorina A, kui võrdluse tulemus on tõene, ja tühi-operaatorina vastupidisel juhul.

Tingimusliku operaatori üldkuju on

if α then A else B ;

kus B võib olla suvaline operaator. Selline tingimuslik operaator töötab võrdluse tõese tulemuse korral operaatorina A ja vastasel juhul operaatorina B.

Tingimuslikus operaatoris pärast sõnu then ja else kirjutatud operaatoreid võib märgendada. Nii võime näiteks kirjutada

```
if x=0 then M1:n:=1 else M2:n:=n+1;
```

Andes suunamisoperaatoriga juhtimise sõnale then järgnevale operaatorile, täidetakse see tingimuslik operaator nii, nagu oleks võrdluse tulemus tõene, sõnale else järgnevale operaatorile juhtimise andmisel aga, nagu oleks võrdluse tulemus väär.

Kui tingimuslikus operaatoris on tarvis täita terve mingi jada operaatoreid, siis tuleb nendest operaatoritest moodustada liitoperaator, s.t. võtta see operaatorite jada operaatorsulgudesse begin ... end. Operaatorsulgudesse tuleb võtta ka sõna then järele paigutatav tingimuslik operaator või tsüklioperaator. Seda arvestades on järgmised tingimuslikud operaatorid kirjutatud õigesti:

```
if x=0 then begin m:=m+1; n:=2×n end else go to B;
if a > b then begin if c=1.0 then stop else go to
K end else go to L;
```

ALGOL-mai keeles ei tohi tingimuslik operaator kokku sisaldada rohkem kui 12 korda sõna if, kusjuures tohib esineda ülimalt kaheksa sõna if, mille vahel ei ole sõna else.[1] Nii näiteks operaator

```
if α1 then begin if α2 then begin if α3 then A1
else A2 end else A3 end;
```

sisaldab kolm sümbolit if, mille vahel ei ole sõna else.

ALGOL-mai keel lubab omistamisoperaatoreis kasutada ka

---

[1] Konstruktsioon while ... do (vt. § 9) samastatakse operaatorite sisalduvuse määramisel konstruktsiooniga if ...then.

nn. tingimuslikke aritmeetilisi avaldisi, s.t. lubatavaks loetakse kirjutis

I := if α then A1 else A2 ;

kus I on muutuja, α võrdlus ning A1 ja A2 aritmeetilised lihtavaldised. Niisuguseks n.ö. tingimuslikuks omistamisoperaatoriks on näiteks

i := if x > y then x + 5 else x - 2 ;

Analoogiliselt on lubatud ka suunamisoperaatorites kasutada tingimuslikke märgendiavaldisi. Vastava n.ö. tingimusliku suunamisoperaatori kuju on järgmine:

go to if α then M1 else M2 ;

kus M1 ja M2 tähendavad märgendiavaldisi varemkirjeldatud tähenduses. Lisaks on lubatud kasutada tingimuslikku suunamisoperaatorit ka kujul

go to if α then (M) else M2 ;

kus M on tingimuslik märgendiavaldis. Näiteks võib programmis kasutada kirjutist

go to if i=k then (if j=k then M else h) else h1;

mis i ja k mittevõrdsuse korral annab juhtimise märgendiga h1 varustatud operaatorile, juhul i=k, j=k operaatorile märgendiga M ja juhul i=k, j≠k operaatorile märgendiga h. Toodud tingimuslikku suunamisoperaatorit võib soovi korral kirjutada ka kahe tingimusliku operaatorina·

if i≠k then go to h1; if j=k then go to M else go to h;

§ 9. TSÜKLIOPERAATOR.

Tsüklioperaatoril saab ALGOL-mai keeles olla kaks väliselt erinevat kuju:

for I := A1, A2 while α do S ;

for I := A2 while α do S ;

kus I on identifikaator (tsükliparameeter), A1 ja A2 aritmeetilised avaldised, α võrdlus ning S operaator. Kui S on siin kas omakorda tsüklioperaator või tingimuslik operaator, siis tuleb ta võtta operaatorsulgudesse.

Vaatleme, kuidas toimub operaatori S korduv täitmine esimese tsüklioperaatori korral. Kõigepealt täidetakse omistamisoperaator I:=A1 ja seejärel S. Pärast operaatori S esmakordset täitmist toimub omistamine I:=A2 ja võrdluse α kontrollimine. Kui võrdluse väärtuseks osutub tõene, siis täidetakse jälle operaator S ja alustatakse uuesti omistamisega I:=A2. Kui võrdluse väärtuseks saadakse väär, siis antakse juhtimine tsüklioperaatorile järgnevale operaatorile.

Toodud kirjeldus iseloomustab ka teise tsüklioperaatori tööd, milles jääb ära tsükliparameetrile algväärtust andev omistamine I:=A1 ja sellele järgnev operaatori S täitmine, s.t. alustatakse kohe omistamisest I:=A2. Erinevus nende kahe tsüklioperaatori vahel seisneb eeskätt selles, et esimeses täidetakse operaator S vähemalt 1 kord, teises võib aga osutuda, et teda ei täideta kordagi.

Esimese tsüklioperaatori tööd võime iseloomustada ka järgneva programmiloigu abil:

```
I := A1; go to M;
M1:I:=A2; if α then M: begin S; go to M1 end;
```

kusjuures teine rida üksinda kirjeldab teise tsüklioperaatori tööd (märgend M on sel juhul muidugi tarbetu).

Illustreerime tsüklioperaatori kasutamist paari lihtsa näitega.

Maksimaalse arvu leidmiseks reaalarvude $a_1, a_2, ..., a_{100}$ hulgast võib koostada näiteks järgmise programmilõigu:

```
R:=a[2]; for i:=4, i+2 while i ≤ 200 do
begin if a[i]>R then R:=a[i] end;
```

(et operaator S on vaadeldaval juhul tingimuslik operaator, siis tuleb ta võtta operaatorsulgudesse).

Teise näitena esitame veel programmilõigu, mis leiab kõikide kolmliikmete $a_i z_j^2 + b_i z_j + c_i$ väärtuste summa (kus $i = 1,2,...,10$; $j = 1,2,...,15$):

```
S:=0.0; for i:=2, i+2 while i ≤ 20 do
begin for j:=2, j+2 while j ≤ 30 do
S:=z[j]× z[j]× a[i]+(b[i]× z[j])+c[i]+S end;
```

Viimases näites on kasutatud tsüklioperaatorit teise tsüklioperaatori sees. Rõhutame, et ALGOL-mai keeles ei tohi programm sisaldada üle kaheksa tsükli tsüklis.

Tsükli täitmise lõpul säilitab tsükliparameeter väärtuse, mille korral võrdluse tulemuseks saadi väär ning parameetri seda väärtust võib järgnevas programmis kasutada. Tsükliparameetri väärtus säilib ka tsüklist väljumisel suunamisoperaatori abil.

Kui tsüklioperaatori koosseisu kuuluvas liitoperaatoris leidub mõni märgendatud operaator, siis saab selle operaatori poole pöörduda ainult samast liitoperaatorist. Siseneda tsüklioperaatorisse suunamisoperaatori abil väljastpoolt ei ole lubatud.

§ 10. KOODOPERAATOR.

ALGOL-mai keel ei võimalda ära kasutada arvuti "Ural-4" kõiki võimalusi. Näiteks puuduvad keeles operaatorid informatsiooni vahetamiseks välismäluga, enamiku välisseadmete poole pöördumiseks ning paljude loogiliste operatsioonide teostamiseks. Kõige selle kompenseerimiseks on ALGOL-mai keeles loodud võimalus esitada mõningaid programmi osi vahetult arvuti käskudes. Selleks tuleb programmi vastavasse kohta kirjutada arvuti käskudesüsteemi käsk, kusjuures iga käsu ette tuleb panna märk ж (tärn) ja käsu järele semikoolon. Mitme sellise käsu üksteisele järgnemise korral võib semikooloni kirjutada vaid viimase käsu järele. Käskudes esitatud programmiosa nimetame koodoperaatoriks ehk operaatoriks "kood". Koodoperaatorit, nagu iga teist operaatorit võib märgendada.

Koodoperaator seostatakse ülejäänud ALGOL-mai programmiga kasutatavate identifikaatorite ning märgendite abil. Nimelt võib koodoperaatori iga käsu ette (enne tärni) kirjutada veel mingi muutuja identifikaatori või mingi operaatori märgendi. Programmi transleerimisel liidetakse käsu aadressosaga vastavale muutujale eraldatud mälupesa aadress või vastava märgendiga määratud programmiosa paiknemise algusaadress.

Toome näitena koodoperaatori algväärtuse omistamiseks täisarvulise massiivi a (mille indeksid algavad nullist) esimesele kümnele elemendile $a_0, a_1, \ldots, a_9$, eeldades, et vastavad väärtused paiknevad juba magnettrumlil nr.0, alates seitsmenda tsooni algusest.

```
TRUM:    a = 51 0000 0
           = 07 0000 0
         a = 00 0011 0
       k Σ = 16 0000 4
         a = 51 0000 4
           = 07 0000 0
         a = 00 0011 0
       k Σ = 14 0000 4
      TRUM = 21 0000 0 ;
```

Paneme tähele, et massiivi identifikaator kirjutatakse käsu ette ilma indekssulgudeta ning käsu aadressosale liidetakse alati massiivi nullindale muutujale vastav aadress (seda isegi siis, kui massiivis puudub element indeksiga null). Kui on tarvis valida teatud indeksiga muutujat, siis tuleb vastav indeks kirjutada käsu aadressiossa. Näiteks kirjeldusega real array a[10:20] määratud massiivi esimese muutuja a_{10} võime tuua summaatorisse koodoperaatoriga a = 42 0024 4; (0024 on indeksi 10 kahekordne väärtus kaheksandarvuna). Analoogiliselt võib tuua kirjeldusega integer array a[-6:8] määratud muutuja a_{-3} summaatorisse koodoperaatoriga a = 01 7775 0; kus vastava käsu saame käsule 02 0000 0 arvu -3 liitmise teel.

Identifikaatoritele eraldatud pesade aadressid ning mär-

genditele vastavad aadressid liidetakse käsu aadressosaga algebraliselt, mida tuleb arvestada miinusmärkidega varustatud käskude moodustamisel. Olgu näiteks tarvis transleerimise tulemusel saada programmi käsk -02 ß 0, kus ß on muutujale b eraldatud pesa aadress. Koodoperaatorit b ≡ -02 0000 0; kasutades saaksime nüüd vale tulemuse ja vea vältimiseks tuleb kasutada näiteks järgmist koodoperaatorit:

```
M:   b ≡ 00 0000 0
       M ≡ 30 0000 0
         ≡-02 0000 0 ;
```

Märgime veel, et koodoperaatori märgi ≡ ees võib kasutada vaid selles programmis juba esinenud märgendeid.

Koodoperaator võimaldab pöörduda ka ALGOL-mai alamprogrammide poole, mis peavad olema vormistatud kujul

M : stop; 01; 02; ...; 0n; go to M;

kus M on alamprogrammi märgend ja 01, 02,...,0n seda alamprogrammi moodustavad operaatorid. Koostatud alamprogrammi poole võib pöörduda kas koodoperaatoriga M ≡ 22 0000 4; või operaatoritega ≡ 30 0307; go to M; Esimest moodust tohib kasutada vaid siis, kui alamprogramm ise paikneb programmis enne seda pöördumist, teist moodust aga sõltumata alamprogrammi ja pöördumise vastastikusest asetusest.

ALGOL-mai keele translaatorit on võimalik kasutada ka koos standardprogrammide koguga "Сторож-44", mis võimaldab lülitada transleeritud programmi kõiki selles kogus leiduvaid programme. Põgusalt vaatleme mainitud kogu kasutamist käesoleva juhendi teises osas, üksikasjaliku kirjelduse võib

leida aga näiteks raamatust: Е.Г. Гавриленко "Автоматизация программирования и стандартные программы для "Урал-2", "Урал-3" и "Урал-4"", Москва, 1966. Standardprogrammide poole saab pöörduda vaid koodoperaatori abil, pöördumise vorm on aga antud vastava standardprogrammi kirjelduses. Siinkohal vaatleme vaid ühte näidet, kus standardprogrammiks olgu Simpsoni valemiga integraali väärtuse leidmise programm. Siinjuures eeldame, et ta on modifitseeritud alates pesast 6000 (modifitseerimist vaata käesoleva juhendi teisest osast). Vastav koodoperaator kirjutatakse kujul

```
      = 22 6000 4
P     = 22 0000 4
delta = 42 0000 4
eps   = 42 0000 4
a     = 42 0000 4
b     = 42 0000 4 ;
```

kus P on integrandiks oleva funktsiooni arvutamise alamprogrammi märgend, delta - integreerimissammu algväärtus, eps - integraali arvutamise absoluutne viga, a ja b vastavalt integraali alumine ja ülemine raja.

Kolm standardprogrammi ja nimelt arvude trükkimise programmid (nr. 525 022 ja nr. 525 016) ning perfokaartidelt arvude sisestamise programm (nr. 525 004) on võetud ALGOL-mai programmoteegi koosseisu ja paiknevad ülesande lahendamise ajal pidevalt arvuti mälus, alates vastavalt pesadest 1150, 1034 ja 1222. Seetõttu võime neid standardprogramme kasutada ilma eelneva modifitseerimiseta, kirjutades ainult pöördumise vastava programmi poole. Tõsi, algselt on mainitud kolm

programmi ette nähtud algandmete sissetoomiseks ja trükkimiseks, kuid neid on ühtlasi mugav kasutada ka transleeritud programmis.

§ 11. PEATUSOPERAATOR JA TRÜKKIMISOPERAATORID.

Peatusoperaator on määratud arvuti töö peatamiseks. Tema kuju on

stop;

Peatuskäsuna tõlgendatakse ka programmi viimast operaatorsulgu end. Operaator stop töötab nagu arvuti "Ural-4" käsk 37 0000, mistõttu pärast operaatori stop täitmist võib jätkata ülesande laherdamist järgmisest operaatorist, vajutades arvuti juhtimispuldil nupule "käivitus".

Tulemuste väljastamiseks on ALGOL-mai keeles kaks trükkimisoperaatorit: print ja out. Nendest esimest tarvitatakse kiirtrükkali, teist laitrükkali poole pöördumisel. Kui tavaliselt lõpetab ALGOL-mai keele operaatori kas end, else või semikoolon (;), siis trükkimisoperaator peab alati lõppema just semikooloniga.

Esimene trükkimisoperaator koosneb sõnalisest põhisümbolist print ja temale järgnevatest (üksteisest komadega eraldatud) aritmeetilistest avaldistest, näiteks

```
print R;
print 3.0,a,b,c,x+5;
print 1, if x > 3.7 then y else z / 3.0-x+y;
```

Tühja rea jätmiseks võib trükitavate avaldiste loetellu lülitada sümboli ␣ (tühik). Vahetult pärast sümbolit print

võib sümboleid ␣ kirjutada kuitahes palju, trükitavate avaldiste loetelus võib aga ainult koma asendada ühe tühikuga, mis tähendab tühja rea trükkimist. Näiteks

```
print ␣ ␣ a,b,c ␣ x+5;
print ␣ 1 ␣ if x > 3.7 then y else z / 3.0-x+y;
```

Trükkimisoperaator ei tohi lõppeda tühikuga. Ainult tühja rea trükkimiseks kiirtrükkalil tuleb kasutada koodoperaatorit = 34 0004;

Operaator out võimaldab aritmeetiliste avaldiste väärtuste trükkimist laitrükkalil. Vastavad väärtused trükitakse liikuva komaga kümnendarvudena (tarvilikud teisendused tehakse automaatselt). See operaator koosneb sõnalisest põhisümbolist out ning temale järgnevast aritmeetiliste avaldiste loetelust, milles avaldised on omavahel komadega eraldatud. Operaatori täitmisel jagatakse väljatrükitavate avaldiste loetelu kaheksakaupa osadeks, millest iga osa moodustab ühe väljatrükitava rea, kusjuures viimane rida võib jääda ka poolikuks.

Tühiku kasutamine operaatoris out on erinev tema kasutamisest operaatoris print. Nimelt tühik tähendab nüüd tühja sümboli viimist väljatrükitava tabeli vastavasse veergu. Pärast sõna out ja pärast iga koma väljastavate avaldiste loetelus on lubatud kirjutada suvaline arv tühikuid. Koma asemele kirjutatud tühikut (nagu operaatoris print) aga tõlgendab translaator nüüd komana. Kui operaatori out aritmeetiliste avaldiste loetelu lõpeb tühikuga (vahetult enne semikoolonit), siis viiakse trükitavate avaldiste väärtused küll trükkimis-

seadme registrisse, kuid registri trükkimist ei teostata. Samas trükitakse ka vea tunnus 22 2105 (vt. lk. 50), mis viitab süntaktilise vea olemasolule, kuna trükkimine jäi ju lõpetamata. Sellist väljundregistrisse viimist on aga mõnikord vaja kasutada niisuguse tabeli ridade moodustamiseks, mille kuju ei saa anda ühe operaatoriga out. Väljundregistri trükkimise võime teostada seejärel kas koodoperaatoriga
= 61 4000 4; või kirjutades suvalise operaatori ette sobivas kohas mingi tähe koos temale järgneva tühikuga, näiteks
A ␣ x:=z; Kui väljundregistrisse ei ole selleks momendiks midagi kantud, siis saame sellisel viisil jätta laitrükkalil tühja rea. Mitme järjestikuse tühja rea jätmiseks võime kasutada mitut järjestikust kombinatsiooni täht-tühik, nii näiteks trükib operaator c ␣ c ␣ c ␣ go to A; kolm tühja rida.

§ 12. KOMMENTAARID.

Erinevalt ALGOL-ist ei kasutata ALGOL-mai keeles sõna comment. Programmi kohta käivaid selgitusi võib aga kirjutada kõikjale sõna end järele. Nimelt pärast sõna end olev sümbolite jada kuni järgneva semikoolonini või üheni sõnadest end ja else jäetakse transleerimisel vaatlusest välja. Näiteks kirjutist

... end esimese osa lõpp; x := 2.7;

tõlgendab translaator samaväärsena kirjutisega
... end; x:=2.7; Sellega seoses tuleb jälgida, et kui meil ei ole tegemist kommentaaridega, siis peab sõna end järel

paiknema kas semikoolon, end või else. Näiteks programmiosas

... end x:=2.7;

ei tõlgenda translaator kirjutist x:=2.7 mitte omistamisoperaatorina, vaid selgitava märkusena.

§ 13. LIITOPERAATOR, BLOKK, PROGRAMM.

Liitoperaator kujutab endast operaatorsulgudesse võetud operaatorite jada. Tema üldkuju on

begin S1; S2; ...; Sn end;

kus S1, S2, ..., Sn on suvalised operaatorid, sealhulgas võivad nad ise olla kas liitoperaatorid või blokid.

Blokk erineb liitoperaatorist vaid selle poolest, et operaatorsulu begin järel leidub vähemalt üks (selles blokis lokaliseeritud) identifikaatorite kirjeldus. Bloki mõiste ongi sisse toodud just identifikaatorite ja märgendite määramispiirkondade täpsustamiseks. Objektid, mille identifikaatorid on kirjeldatud bloki alguses, on selles blokis lokaliseeritud ja neid ei saa väljaspool blokki samas tähenduses kasutada. Samuti on kõik bloki koosseisu kuuluvate operaatorite märgendamiseks kasutatud märgendid automaatselt (ilma mingi täiendava kirjelduseta) lokaliseeritud kõige väiksema mahuga blokis, milles see märgendamine toimus.

Programmi on tarvis jaotada blokkidesse eeskätt vaid juhul, kui ühe blokina kirjutatult ületaks programm mälu mahu.

ALGOLis kirjutatud programm peab olema vormistatud bloki-

na. ALGOL-mai keeles peab täiendavalt arvestama, et programmi ei tohi märgendada. Peale selle jagunevad programmi alguses toodud kirjeldused kahte ossa: algandmete kirjeldused ja nende muutujate kirjeldused, millele omistatakse väärtused alles programmi täitmise käigus.

Programmis kasutatavad algandmed tuuakse arvutisse pärast transleerimise lõppu automaatselt ning omistatakse vastavatele muutujatele, eeldades, et nende sisendseadmes paiknemise järjekord ühtib kirjeldamisjärjekorraga. Seetõttu peavad algandmed olema kirjeldatud (ja nende väärtused perforeeritud) rangelt kindlaksmääratud järjekorras: kõigepealt reaalarvude tüüpi lihtmuutujad, siis reaalarvude tüüpi massiivid, siis täisarvude tüüpi lihtmuutujad ja lõpuks täisarvude tüüpi massiivid. Vastavat tüüpi algandmete puudumisel puuduvad ka vastavad kirjeldused (sama tüüpi lihtmuutujate ning massiivide kirjelduste järjekorda on siiski lubatud ka vahetada). Ülejäänud muutujate ja lülitite kirjeldused võivad vahetult algandmete kirjelduste järel paikneda juba suvalises järjekorras. Mujal programmis saab sisendite poole pöörduda vaid vastavate standardprogrammide või koodoperaatori abil.

ALGOL-mai keeles kirjutatud programm ei tohi olla pikem kui 3000_8 pesa, s.t. ei tohi olla pikem kui 3072 sümbolit. Blokis samaaegselt kehtivate identifikaatorite ja märgendite arv ei tohi olla suurem kui 150. Blokis ei tohi olla üle neljakümne pöördumise niisuguste märgendite poole, mis on kirjutatud programmis hiljem kui suunamisoperaator sellele märgendile.

§ 14. PSEUDOMÄRGEND ОТЛАД.

Pseudomärgendit ОТЛАД kasutatakse niisuguste operaatorite märgendamiseks, mis tulevad täitmisele ainult võtme 7 sisselülitatuse korral. Seda operaatorit on otstarbekohane kasutada programmide silumisel, näiteks lisades mõningaid vahetulemusi trükkivaid operaatoreid:

```
отлад : print ␣ a;
отлад : begin a := 1; for i := 1, i+1
        while i ≤ n do print x[i]; end;
```

Erinevalt tavalistest märgenditest võib pseudomärgendit ОТЛАД kasutada ühe bloki piires suvaline arv korda. Pseudomärgendit ОТЛАД ei tohi kasutada suunamisoperaatoris.

II. TRANSLAATORI KASUTAMINE

§ 1. ÜLDISI ANDMEID TRANSLAATORIST.

ALGOL-mai programmide transleerimiseks peab translaator asuma magnettrumlil nr. 0, kust ta iga üksikülesande jaoks tuuakse uuesti sisemällu. Transleerimise ajal asuvad arvuti sisemälus nii ALGOL-mai programm kui ka translaatori transleeriv osa. Selline paigutus annab küll suure transleerimiskiiruse, kuid ühtlasi piirab transleeritava programmi pikkust. Nimelt sisestatakse ALGOL-mai programm alati alates pesast 4100 ning ta võib võtta enda alla mälu ülimalt kuni pesani 7100. Arvestades, et ühte topeltpessa mahuvad nelja ALGOL-mai sümboli koodid, saame programmi maksimaalseks lubatavaks pikkuseks 67 perfokaarti.

Pärast transleerimist jäävad arvuti mällu vaid koostatud programm (mis paigutatakse vastavalt programmeerija poolt etteantud algusaadressile) ja ALGOL-mai standardfunktsioonide kogusse kuuluvad programmid, mis asuvad pesades 0052 - 1432.

Translaator võtab magnettrumlil enda alla neljanda ja viienda tsooni, tsoonid 0 - 3 on jäetud standardprogrammide kogu "Сторож-44" jaoks. Nii translaatori kui ka "Сторож-44" magnettrumlile viimise eest hoolitseb operaator. Programmeerijal tuleb aga iga ülesande jaoks koostada lahendamisjuhend, milles muidugi kõigepealt öeldakse, et programm kasutab

ALGOL-mai translaatorit. Peale selle on tarvis näidata võtmete seis arvuti juhtimispuldil (eraldi transleerimise ja lahendamise jaoks) ning kasutatavate "Сторож-44" programmide numbrid.

§ 2. KAARDIPAKK.

Translaatori kasutamiseks tuleb moodustada järgmistest osadest koosnev kaardipakk (osad paiknevad toodud järjekorras):

1) perfokaart translaatori toomiseks magnettrumlilt sisemällu,
2) perfokaart alginformatsiooniga,
3) perfokaardid ALGOL-programmiga,
4) perfokaardid informatsiooniga standardprogrammide toomiseks sisemällu,
5) perfokaardid reaalarvudena kirjeldatud algandmetega,
6) perfokaardid täisarvudena kirjeldatud algandmetega.

Perfokaart translaatori toomiseks on iga ülesande korral ühesugune, tema esimesse kuude ritta perforeeritakse kaheksandarvud

51 0020 0104 000
00 7777 0220 120
25 7754 6557 771
24 0011 0317 775
21 0004 0456 001
-00 4003 0446 152 ,

ülejäänud viide ritta aga nullid. Selle kaardi lisab pakile

tavaliselt operaator (kui juhendis on seda nõutud).

Alginformatsiooniga perfokaart annab ette mälujaotuse. Selleks peab tema esimesse viide ritta olema perforeeritud kaheksandarvud

$$
\begin{array}{llll}
00 & a_1 & 0000 & 000 \\
00 & a_2 & 0000 & 000 \\
00 & b_1 & 0000 & 000 \\
00 & b_2 & 0000 & 000 \\
00 & 0000 & 0000 & 071\ ,
\end{array}
$$

kus

a_1 on pesa aadress, millest alates peab translaator paigutama transleeritud programmi ($a_1 \geq 1500$). Programmi paigutamisel mälus tuleb arvestada, et temas kasutatud konstandid kirjutatakse vahetult enne transleeritud programmi ja paigutatakse aadresside vähenemise suunas alates aadressist $a_1 - 1$. Näiteks kui meil on fikseeritud $a_1 = 1500$ ja esimeseks konstante sisaldavaks operaatoriks programmis osutub a := 2, siis paigutatakse arv 2 pessa 1477. Tuleb jälgida, et konstandid ei satuks pesadesse, mille aadressid on väiksemad kui 1432 (need on varutud ALGOL-mai standardprogrammide ja -protseduuride jaoks);

a_2 on pesa aadress, milleni on lubatud paigutada transleeritud programmi. Selline piiramine kindlustab sisemälus koha algandmete ja modifitseeritavate standardprogrammide jaoks. Aadress a_2 tuleb valida nii, et oleks $a_1 < a_2 < 7000$;

b_1 on pesa aadress, millest alates translaator paigutab reaalarvulised algandmed (täisarvud paigutatakse vahetult reaalarvude järele ja nende algusaadressi ei ole tarvis näidata);

b_2 on aadress, mis näitab tööpesade lubatavat lõppu ja ta tuleb valida nii, et oleks $b_1 < b_2 < 7732$.

Mälu jaotuses ei tohi vahemikud $[a_1,a_2)$ ja $[b_1,b_2)$ lõikuda ei omavahel ega ka standardprogrammide jaoks eraldatud vahemikuga. Aadressid a_1,a_2,b_1 ja b_2 peavad olema paarisarvud.

ALGOL-mai programm kodeeritakse vastavalt ALGOL-mai koodi tabelile (vt.lk. 55), kusjuures igale keele elemendile vastab kolmekohaline kood. Iga neli koodi moodustab ühe rea perfokaardil (rea viimane kaheksandkoht jääb kasutamata) ja salvestatakse ühte topeltpesasse. Kodeerimisel tuleb kinni pidada järgmistest reeglitest.

Kui tärni (*) kood ei ole viimane kood topeltpesas, siis sellesse pessa kirjutatakse tärni järele nullid.

Tärni koodi sisaldavale topeltpesale järgnevasse lühipessa kirjutatakse arv või käsk selliselt, nagu ta on kirjutataud ALGOL-mai programmis. Järgnev lühipesa jääb tühjaks.

ALGOL-mai programmi lõputunnuseks kirjutatakse arv 777 777 777 vahetult programmi viimase operaatorsulu end järele ja lõputunnusena veel ühte topeltpessa arv 00 0000 0000 071. Näiteks ALGOL-mai programm

```
begin real x ; integer t, p ;
      real y ; y := sin (x)↑2 ;
      print y ; t := t + p ;
         ≡ 22 1034 4
      y  ≡ 00 0000 0
         ≡ 00 0002 0
         ≡ 00 0001 0 ;
end
```

näeb kodeeritult välja järgmiselt (numbrite rühmitamisel on arvestatud TRÜ Arvutuskeskuses kasutusel olevat blanketti, mitte jaotust koodide kaupa):

```
60 2607 3256 050 - begin real x ;
61 0222 4013 200 - integer t, p
60 5607 3236 050 - ; real y ;
32 3402 2212 100 - y := si
21 5405 3254 060 - n (x)
41 2102 6056 140 - ↑ 2 ; print
32 3605 2224 020 - y ; t :=
22 2404 3206 050 - t + p ;
42 3000 0000 000 - ≡
22 1034 4000 000 - 22 1034 4
32 3423 0000 000 - y ≡

00 0000 0000 000 - 00 0000 0
42 3000 0000 000 - ≡
00 0002 0000 000 - 00 0002 0
42 3000 0000 000 - ≡
00 0001 0000 000 - 00 0001 0
60 5604 7777 770 - ; end 777 777
77 7000 0000 000 - 777
00 0000 0000 071 - lõputunnus
```

Perfokaardid informatsiooniga standardprogrammide toomise ja paigutamise kohta sisemälus on kaardipakis vaid sel ju-

hul, kui transleeritud programm peab sisaldama programme kogust "Сторож-44". Vastava kogu kasutamise või mittekasutamise määrab võtme 5 asetus transleerimisel (vt. § 3). Üksikasjaliku ülevaate selle standardprogrammide kogu ja vastava koostava programmi (ССП-42) kohta annab leheküljel 28 nimetatud raamat, millega mainitud kogu kasutamiseks tingimata tuleb tutvuda. Selles raamatus kirjeldatud üldiste põhimõtete tundmist eeldades piirdume siin vaid paari vahetult translaatoriga seotud märkusega.

Standardprogrammide toomise ja paigutamise kaart sisaldab kahte liiki informatsiooni: fiksaatorid ja standardprogrammide numbrid. Fiksaator kujutab endast selle pesa aadressi, millest alates tuleb paigutada sissetoodav standardprogramm. See aadress a kantakse perfokaardi ühte ritta kujul

- 00 0000 00 a 7 .

Kasutatava standardprogrammi number b (kuuekohaline kaheksandarv) perforeeritakse kahekordselt ning igale numbrile tuleb eraldada kaks rida perfokaardil, mis täidetakse järgmiselt:

```
-      b     b     1
0 00 0000  0000  006 .
```

Kui transleeritud programmis kasutatakse mitut standardprogrammi, siis võib fiksaatori abil fikseerida kas iga programmi, teatud programmide või siis ainult esimese programmi algusaadressi. Fikseerimata aadressidega programmid paigutab koostav programm (ССП-42) vahetult eelmiste järele, trükkides ühtlasi laitrükkalil vastavad aadressid. Kaardipaki vaa-

deldava osa lõputunnuseks on tühi perfokaart.

Kui näiteks on tarvis tuua sisemällu programmid numbritega 525 021 ja 000 370 ning paigutada nad alates vastavalt pesadest 5000 ja 6000, siis kirjutatakse vastav informatsioon järgmiselt:

```
- 00 0000 0050 007
- 52 5021 5250 211
  00 0000 0000 006
- 00 0000 0060 007
- 00 0370 0003 701
  00 0000 0000 006 .
```

Kui aga on tarvis tuua sisemällu programmid 525 023 ja 000 360,alates pesast 6210, ning programm 000 430,alates pesast 5150, siis vastava informatsiooni võib esitada nii:

```
- 00 0000 0062 107
- 52 5023 5250 231
  00 0000 0000 006
- 00 0360 0003 601
  00 0000 0000 006
- 00 0000 0051 507
- 00 0430 0004 301
  00 0000 0000 006 .
```

Kui standardprogrammid sisaldavad pöördumisi teiste standardprogrammide poole, siis toob koostav programm automaatselt mällu ka need standardprogrammid, mille poole pöördutakse.

Algandmetena sisestatavad reaalarvud perforeeritakse liikuva koma kümnendarvudena. Vahetult viimase arvu järele

perforeeritakse lõputunnuseks arv $0 \cdot 10^{39}$. Reaalarvulise massiivi puudumisel tuleb kaardipaki vastavale kohale asetada perfokaart, mille esimeses reas on arv $0 \cdot 10^{39}$.

Algandmetena sisestatavate täisarvude perforeerimine on täiesti analoogiline reaalarvude perforeerimisega.

Kui ülesandes puuduvad nii reaal- kui ka täisarvudena kirjeldatud algandmed, siis on täisarvude lõputunnuseks arvud 0 ja $0 \cdot 10^{39}$, mis perforeeritakse toodud järjekorras kahte järjestikusesse topeltpessa.

Koostatud perfokaartide pakk viiakse arvutisse algkäivitusega. Ülesande süntaktiliseks silumiseks võib kaardipakk koosneda vaid osadest 1), 2) ja 3). Sel korral on süntaktilise silumise lõpp-peatus (pesas 7740) kogu töö lõpp-peatuseks.

§ 3. VÕTMED.

Translaatori kasutamisel tuleb eristada võtmete asetust transleerimisel nende asetusest koostatud programmi täitmisel.

Võtmete 1 – 7 asetuste tähendused transleerimisel on toodud järgnevas tabelis.

Võtme nr.	Võtme seis	
	ülal (sees)	all (väljas)
1		Kiirtrükkal trükib transleeritud programmi.
2		Kiirtrükkal trükib programmi ALGOL-mai koodis kaartide kaupa.
3	Laitrükkal trükib ALGOL-mai programmi kujul 1.	
4	Väljundperforaator väljastab transleeritud programmi.	
5	Trumlilt tuuakse koostav programm ССП-42 ja antakse juhtimine sellele.	
6	Laitrükkal trükib transleeritud programmi.	
7	Laitrükkal trükib ALGOL-mai programmi kujul 2.	

ALGOL-mai programmi on võimalik lasta trükkida kahel erineval kujul. Kuju 1 korral arvestatakse programmi blokilist struktuuri: iga sõna begin, end, if, for ja koodoperaatori märk * alustab uue rea. Teineteisele vastavad operaatorsulud begin ja end trükitakse kohakuti. Programmis esinevad märgendid eraldatakse lehe vasakule äärele. Kuju 2 kasutamisel trü-

kitakse programm nii, nagu ta on jaotatud perfokaartidel.

Väljatrükkimisel kasutatakse ALGOL-mai sõnadele vastavaid venekeelseid termineid. Eri tähestike ja suurte ning väikeste tähtede eraldamiseks trükitakse ladina ja kreeka tähestiku väikestele tähtedele alla märk ↑, ladina ja kreeka tähestiku suurtele tähtedele märk → ning vene tähestiku väikestele tähtedele märk ✳. Näiteks ladina tähestiku tähe b asemel trükitakse Б↑, L asemel Л→ ja vene tähestiku tähe з asemel З✳. Kreeka tähtede ning mõningate ladina tähtede trükkimisel kasutatakse nende nimetusi, kusjuures suuri ja väikesi tähti eraldavad ka siin märgid alumisel real. Näiteks α asemel trükitakse АЛЬФА↑↑↑↑↑, Ω asemel ОМЕГА→→→→→, j asemel ЙОТ↑↑↑ jne. Ladina tähestikus kirjutatud teksti trükkimisel kasutatakse ka lühendeid: ja asemel Я, ju asemel Ю ning jo asemel Ё. Laitrükkali tähtede hulgas puuduva vene tähestiku tähe ъ asemel trükitakse ь, puuduvate märkide ␣, $_{10}$, ≥ ja ≤ asemel kasutatakse aga kirjutisi →, ↑✳, ≧ ja ≦. ALGOL-mai programmide trükkimisel kontrollib arvuti vaid kasutatud koodide lubatavust ning iga vale koodi korral trükib (Н.К.α), kus α on programmis kasutatud vale kood. Kogu programmi väljatrükkimine lõpeb arvuga 777.

Transleeritud programmi väljastamisel perfokaartidele perforeerib väljundperforaator esimesele kaardile arvu

$$00\ a_1\ 00\ a_2\ 0\ ,$$

kus a_1 on programmi algusaadress ja a_2 kontrollsumma aadress.

Koostav programm ССП-42 ei ole lülitatud vahetult translaatori koosseisu ning tema kasutamiseks peab ta asuma

"Сторож-44" programmina magnettrumlil.

Võtmete igasuguse asetuse korral väljastab kiirtrükkal viimase ALGOL-mai programmi poolt hõivatud pesa aadressi, alginformatsiooni kaardi sisu, informatsiooni vigadest ja peatustest (vt. § 4), programmi sisenemise aadressi, programmi alg- ja lõppaadressid ning transleeritud programmi kontrollsumma.

Transleeritud programmi töötamise ajal määrab võtme 7 seis pseudomärgendiga ОТЛАД märgendatud operaatorite täitmise või mittetäitmise. Teiste võtmekäskude programmi lülitamiseks tuleb kasutada koodoperaatorit.

§ 4. TRÜKKIMINE TRANSLEERIMISEL.

Lisaks ALGOL-mai programmi ja transleeritud programmi (eelmises paragrahvis kirjeldatud) väljastamise võimalustele töötab kogu transleerimise ajal veel kiirtrükkal, mis väljastab teateid programmis esinevatest süntaktilistest vigadest ja peatusoperaatoritest.

Kui translaator leiab ALGOL-programmis süntaktilise vea, siis jätab ta vahele kogu informatsiooni alates veast kuni järgmise semikoolonini või sõnani begin, if, for, go to, end, stop, print. Samal ajal trükib kiirtrükkal välja vea liigi ja vahelejäetud informatsiooni sisaldavate pesade numbrid. Teade vea kohta trükitakse kas kujul 22 α, ß, $\gamma_1, \ldots, \gamma_n$ või kujul 22 α, 25 n_1, n_2, ß, kus 22 on vea esinemise tunnus, α - vea liiki määrav kood (vt. § 6), ß - selle pesa aadress, millesse salvestatud informatsioonis oli viga, γ_i - nende pe-

sade aadressid, milles olev eraldav kood (nendeks on koodid 401,...,420,422,424,...621) ja temale eelnev informatsioon jäetakse transleerimisel vahele ning n_1 ja n_2 - vastavalt tsükliloendaja ja summaatori seis vea esinemise momendil. Kui näiteks teade süntaktilisest veast trükitakse kujul:

```
22 0755
00 4166
00 4166
00 4170
00 4174 ,
```

siis vastavalt vigade dešifreerimise tabelile (vt. § 6) näeme, et programmis on kasutatud kirjeldamata identifikaatorit (α=0755), mis asub pesas 4166, ja vea tõttu jäeti vaatlusest välja programmilõik kuni pesa 4174 esimese eraldava koodini.

Iga ALGOL-mai programmis esineva operaatori stop korral kirjutatakse transleeritud programmi vastavale kohale käsk 37 a 0, kus a on sedasama käsku sisaldava pesa aadress. Ühtlasi trükitakse vastav käsk kiirtrükkalil. Peatuskäskude järjekord kiirtrükkali lindil vastab operaatorite stop esinemise järjekorrale ALGOL-mai programmis.

Vahetult pärast transleerimise lõppu loeb arvuti perfokaartsisendilt algandmed sisemällu ja trükib nad kontrollimiseks laitrükkalil.

§ 5. TRANSLEERITUD PROGRAMMIDE UUESTI KASUTAMINE.

Transleerimise lõppedes jääb arvuti seisma ning käsuregistrisse tuuakse käsk 22 a 0, kus a on transleeritud programmi sisenemise aadress. Ülesande lahendamiseks tuleb arvu-

ti uuesti käivitada.

Translaatori suurt töökiirust arvestades võib lühemaid ning harva vaja minevaid programme transleerida ka igal kasutamisel uuesti. Pikemate ning sageli tarvitatavate programmide korral ei ole see aga otstarbekas. Translaatoriga koostatud programmide juures tuleb pidada silmas, et nad on translaatoriga tihedalt seotud, ja ilma uuesti transleerimata saab neid kasutada vaid siis, kui translaator asub magnettrumlil. Et translaator väljastab perfokaartidele ainult transleeritud programmi, jättes väljastamata ССП-42 abil modifitseeritud standardprogrammid, siis viimaste esinemise korral tuleb nad modifitseerida uuesti vahetult ССП-42 ja süsteemi "Сторож-44" abil.

Varem transleeritud programmide kasutamisel tuleb koostada järgmistest osadest koosnev kaardipakk:

a) Standardne perfokaart ALGOL-mai biblioteegi toomiseks magnettrumlilt operatiivmällu:

51 0034 0100 000
00 1517 0400 001
25 1460 6543 231
24 0011 0303 235
21 0004 0443 056 .

b) Reaalarvuliste algandmete algusaadress, mis on perforeeritud lühikese pesa arvuna kaardi esimesel real (kaardi teised read peale kontrollsumma peavad olema nullid).

c) Reaal- ja täisarvulised algandmed selliselt, nagu nad olid ALGOL-mai programmis.

d) Väljundperforaatoril väljastatud lahendusprogramm koos väljastatud algkaardiga.

e) Lõputunnusena arv 0.10^{39}, perforeerituna eraldi kaardile.

f) Kaart juhtimiskäsuga 22 a 0, kus a on programmi sisenemise aadress.

Ettevaatlik tuleb siinjuures olla programmide ja algandmete kokkusobitamisel. Et programmi lõputunnus on eraldi kaardil, siis arvestab arvuti programmi kuuluvaiks ka tühjad pesad programmi viimasest käsust kuni lõputunnuseni (maksimaalselt 10 pesa). Tuleb jälgida, et arvuti ei kannaks sellega nulle algandmete jaoks määratud pesadesse. Sellise olukorra tekkimisel tuleb ümber perforeerida programmi viimane kaart, kirjutades lõputunnuse vahetult programmi viimase käsu järele.

§ 6. VIGADE DEŠIFREERIMINE.

Leides programmis süntaktilise vea, trükib arvuti vastava teate ja jätkab transleerimist. Toome järgnevalt vigade tunnused koos nende tähendustega.

22 0321 Programmis esineb ALGOL-mai keeles mitte ettenähtud kood.

22 0535 Massiivikirjelduses massiivi identifikaatorile ei järgne ei [ega koma.

22 0556 Massiivi pikkus on negatiivne (alumine raja on suurem kui ülemine).

22 0617 Massiivikirjelduses kas ülemise raja järel puudub] või siis] järel pole ei koma ega semikoolonit.

22 0621 Massiivikirjelduse järel puudub semikoolon.

22 0651 Viga lülitikirjelduses.

22 0740 Identifikaator või märgend on kirjeldatud blokis kaks korda. Operaatoris puudub identifikaator.

22 0755 Programmis on kasutatud kirjeldamata identifikaatorit.

22 1044 Suunamine on tehtud identifikaatorile, mis ei ole märgendiks.

22 1053 Suunamisoperaatoris seisab märgendi järel vale sümbol (lubatavateks on vaid semikoolon, end, else ja lõpetav ümarsulg).

22 1121 Suunamisoperaatoris puudub märgend.

22 1160 Viga tingimuslikus operaatoris (sõnale else ei vasta sõna if).

22 1237 Viga operaatoris print.

22 1261 Viga täisarvu kirjutamisel. Identifikaator aritmeetilises avaldises algab numbriga.

22 1500 Märgi [ees seisab identifikaator, mida ei ole kirjeldatud lüliti või massiivi identifikaatorina.

22 1506 Indekssulgudes on reaalarvude tüüpi muutuja identifikaator.

22 1515 Märgi] järel seisab identifikaator või arv.

22 1535 Puudub]. Indeksavaldises on märkidest + ja - erinevaid sümboleid.

22 1565 Omistamisoperaatori vasakute poolte loetelu sisaldab nii täisarvulisi kui ka reaalarvulisi muutujaid.

22 1621 Viga omistamisoperaatoris.

22 1630 Puudub süntaksi järgi vajalik sõna end, else või semikoolon (võimalik, et eespool olnud vea tõttu on vaatlusest välja jäetud).

22 1665 Viga koodoperaatori kasutamises. Operaator algab sümboliga := , [või ␣ ja tema ees ei ole identifikaatorit.

22 1727 Tsükli päises puudub while. Aritmeetilises avaldises esineb koma.

22 2036 Tingimuslikus aritmeetilises avaldises puudub else.

22 2040 Operaator on valesti alustatud.

22 2070 Sõnale then või do järgneb vahetult tingimuslik operaator.

22 2105 Viga aritmeetilises lihtavaldises.

22 2142 Ümarsulgude ees seisab funktsiooni nimest erinev identifikaator.

22 2235 Ümarsulgudesse ei ole midagi kirjutatud.

22 2252 Lõpetava ümarsulu järel seisab identifikaator.

22 2326 Aritmeetilises avaldises ei ole tehtemärgi ees identifikaatorit.

22 2336 Aritmeetilises avaldises ei ole tehtemärgi ees identifikaatorit.

22 2334 Aritmeetilises avaldises esineb vale tüüpi identifikaator.

22 2442 Täisarvulise jagamise operandiks on reaalarvude tüüpi avaldis.

22 2545 Operaatoris jäid loendajad tühjendamata.

22 2621 Suunamine identifikaatorile.

22 2731 Programmis esineb ALGOL-mai keeles mitte ette nähtud kood.

22 2736 Programmis jäid loendajad tühjendamata.

Kujul 25α, 25n_1, n_2, ß trükitakse teade vea kohta välja vaid juhtudel α=2545 ja α=2736. Need vead tähendavad, et (vastavalt kas operaatori või kogu programmi) transleerimise lõpuks pole mingi loendaja jõudnud nullseisu. Tühjendamata jäänud loendaja väärtuse näitab summaatori väljatrükitav sisu n_2, loendaja nime aga määrab tsükliloendaja väljatrükitav sisu n_1 järgmise tabeli kohaselt:

n_1 väärtus	tühjendamata loendaja nimi	
	α=2545 korral	α=2736 korral
0000	Φ	a_m
0001	$\sigma_{go\ to}$	a_{for}
0002	σ_a	a_{else}
0003	$a_{(k}$	a_{if}
0004	puudub	Φ

Loendajate σ_a ja $\sigma_{go\ to}$ väärtused (aadressiühikutes) näitavad niisuguste tingimuslike avaldiste arvu, milles pole veel esinenud sõna else.

Loendajate $a_{(k}$ ja Φ väärtused tähendavad alustavate ning lõpetavate ümarsulgude arvu mittevastavust. Mittevastavuse suurust näitab summaatori sisu: erinevusele ühe sulu võrra vastab loendaja $a_{(k}$ korral üks aadressiühik ja loendaja Φ korral kaks aadressiühikut.

Loendajad a_{if}, a_{else} ja a_{for} näitavad lõpetamata tingimuslikke või tsüklioperaatoreid. Igale lõpetamata operaatori-

le vastab kaks aadressühikut summaatoris. Kui näiteks väljatrükitud teates vea kohta on n_2=0004 ja n_1=0001, siis see ütleb, et programmis leidub kaks lõpetamata tsüklit.

Loendaja a_m näitab niisuguste suunamisoperaatorite arvu, mida ei saa täita vastavate märgendite puudumise tõttu. Igale veel kirjeldamata märgendile vastab neli aadressiühikut summaatoris.

Kui näiteks trükitakse välja teade vea kohta kujul:

22 2545
25 0003
00 0001
00 4175 ,

siis see tähendab, et viga selgus pesas 4175 lõppeva operaatori analüüsimisel. Tsükliloendaja sisu ütleb, et operaatoris ei ole vastavust ümarsulgude vahel, ja summaatori sisust järeldub, et erinevus on ühes sulus.

Süntaktiliste vigade parandamisel on kasulik arvestada asjaolu, et iga eraldava koodi ees võib olla suvaline arv nulle. Nii näiteks võime veaga perfokaardi tarviduse korral asendada kahe kaardiga, milles osa ridu võivad olla nullid.

§ 7. AVARIIPEATUSED TRANSLEERIMISEL.

Eelmises paragrahvis loetletud süntaktilises vead ei tee võimatuks edasist transleerimist (kuigi saadav programm pole enamasti töötamiskõlbulik). Kõrvuti nendega võivad aga programmis esineda vead, mille puhul transleerimise jätkamine osutub võimatuks. Sel korral katkestatakse transleerimi-

ne ja programm tagastatakse koostajale. Vea iseloomu määrab nüüd selle pesa aadress, milles paikneva käsu toimel translaator peatus. Võimalike peatuste põhjused on järgmised.

Peatus pesas 0460. Ühes avaldises leidub rohkem kui 40 samaaegselt avatud sulgu.

Peatus pesades 0516 või 2207. Tööpesade jaoks eraldatud mäluosa ületäitumine.(Meenutame, et tööpesad programmile eraldab translaator vahetult muutujate jaoks määratud pesade järele ja lubatav mälu määratakse kindlaks algkaardiga.) Vea parandamiseks tuleb muuta mälujaotust.

Peatus pesas 0737. Identifikaatorite arv ületab lubatava.

Peatus pesas 1072. Transleeritud programmi kandumine ALGOL-mai programmi peale. ALGOL-mai programm paigutatakse mällu alates pesast 4100, transleeritud programmi hakatakse paigutama alates pesast 4000. Peatuse tekkimine ütleb, et transleeritud programm kasvab kiiremini, kui jõutakse analüüsida ALGOL-mai programmi. Vea kõrvaldamiseks tuleb ALGOL-mai programm kirjutada korrektsemalt (näiteks jälgides, et reaalarvulistele muutujatele omistataks reaalarvulisi väärtusi jne.) või lisada ALGOL-mai programmi mõni tühi perfokaart (muidugi lubatavasse kohta!).

Peatused pesades 1077 või 3064. Programm ulatub algandmete jaoks eraldatud pesadesse. Vea parandamiseks tuleb muuta mälujaotust.

Peatused pesades 1375, 1463 või 1677. Need peatused tekivad juhul, kui informatsioonid veel kirjeldamata märgenditest ja konstantidest lõikuvad (esimene kirjutatakse aadresside kasvamise, teine kahanemise suunas). Vea parandamiseks

tuleb programmi muuta selliselt, et väheneks suunamiste arv veel kirjeldamata märgenditele või võtta osa konstante niisugusteks muutujateks, milledele antakse väärtus algandmete omistamise käigus.

Peatus pesas 3137. Programm ei alga sõnaga begin.

Peale avariipeatuste transleerimisel võivad tekkida ka avariipeatused transleeritud programmide täitmisel. Kõik sellised avariipeatused tekivad pesas aadressiga 1025, kusjuures vea liik määratakse kiirtrükkalil trükitud tunnuse

+ + + + + + + + + + + 0 0 0
0 3 7 0 0 0 0 0 0 0 0 α
0 2 2 ß 0 0 0 0 α
+ + + + + + + + + + + 0 0 0

järgi: juhul α=200 on programmis tegemist ruutjuure leidmisega negatiivsest arvust, juhul α=460 osutub eksponentsiaalfunktsiooni argument suuremaks kui 43, 67 ja juhul α=560 on logaritmfunktsiooni argument mittepositiivne. Tunnus ß näitab koha transleeritud programmis, kus olev käsk andis juhtimise vastava funktsiooni arvutamise programmi.

ALGOL-mai koodi tabel

Vene tähed				Ladina tähed				Kreeka tähed				Sümbolid		Reserveeritud sõnad	
А	340	а	300	A	340	a	300	Α	340	α	200	,	401	if	601
Б	341	б	301	B	342	b	201	Β	342	β	241	:=	402	begin	602
В	342	в	302	C	361	c	321	Γ	343	γ	240	[	403	else	603
Г	343	г	303	D	244	d	204	Δ	274	δ	234	]	420	end	604
Д	344	д	304	E	345	e	305	Ε	345	ε	235	+	404	;	605
Е	345	е	305	F	264	f	224	Ζ	247	—	—	(	405	array	606
Ж	346	ж	306	G	246	g	206	Η	355	η	236	)	406	real	607
З	347	з	307	H	355	h	225	Θ	272	ϑ	232	×	407	integer	610
И	350	и	310	I	250	i	210	Ι	250	ι	210	-	410	switch	611
Й	351	й	311	J	251	j	211	Κ	352	æ	276	/	411	go to	612
К	352	к	312	K	352	k	312	Λ	243	λ	203	↑	412	for	613
Л	353	л	313	L	253	l	213	Μ	354	μ	254	÷	413	print	614
М	354	м	314	M	354	m	214	Ν	255	ϑ	237	=	414	stop	615
Н	355	н	315	N	255	n	215	Ξ	267	ξ	227	≠	415	then	616
О	356	о	316	O	356	o	316	Ο	356	ο	316	:	416	do	617
П	357	п	317	P	360	p	320	Π	357	π	277	⌴	417	while	620
Р	360	р	320	Q	252	q	212	Ρ	360	ϱ	230	.	421	out	621
С	361	с	321	R	260	r	220	Σ	271	σ	231	10	422		
Т	362	т	322	S	261	s	221	Τ	362	τ	262	≠	423		
У	363	у	323	T	362	t	222	Υ	273	υ	233	≥	424		
Ф	364	ф	324	U	263	u	223	Φ	364	φ	270	<	425		
Х	365	х	325	V	242	v	202	Χ	365	χ	265	≤	426		
Ц	366	ц	326	W	245	w	205	Ψ	257	ψ	217	>	427		
Ч	367	ч	327	X	365	x	325	Ω	256	ω	216	0	100		
Ш	370	ш	330	Y	363	y	323					1	101		
Щ	371	щ	331	Z	247	z	207					2	102		
Ь	372	ь	332									3	103		
Ъ	373	ъ	333									4	104		
Ы	374	ы	334									5	105		
Э	375	э	335									6	106		
Ю	376	ю	336									7	107		
Я	377	я	337									8	110		
												9	111		

S I S U K O R D

Hind 15 kop.