nélkül. Az eljárások formális paramétereit az eljárás deklarációjakor fel kell sorolni és azok egyéb attributumait az eljáráson belüli deklarációval kell megadni.
Részenkénti fodithatóság. A nyelv megengedi a főprogramon kivüli önálló eljárások használatát. A szegmensben használt, a főprogramban deklarált változókat EXTERNAL jelöléssel kell ellátni. Ezeknek a fordító nem foglal helyet csak nevüket beteszi az ESD (External Symbol Dictionary)-be. A külön forditott részek összekapcsolását szerkesztő végzi.

4.3.2 IMP

Az IMP nyelvet az "Edinburgh Regional Computing Centre"-ben fejlesztették ki, az első Language Manual 1970-ben jelent meg. Azóta több gépre implementálták, többek között az ICL 4/75-re és az IBM 370/158-ra. Magyarországon az Országos Tervhivatal Számitóközpontjában honositották és használják.

Nagyon erős ALGOL-60 befolyás látszik a nyelven. A nyelv jellemzője, hogy mind adatszerkezete, mind vezérlési strukturái igen választékosak, azonban nem mindig a legmodernebb nyelvi eszközöket tartalmazzák. Legjobb példa erre a többirányu elágaztatást megvalósitó switch, amelyet a modern nyelvekben szinte teljesen kiszoritott a CASE utasitás.

Minden kulcsszó % jellel kezdődik, igy a kulcsszavak nem fenntartott nevek.

4.3.2.1 Az IMP adatszerkezete

Az IMP adatai lehetnek: aritmetikai változó (szám), string, pointer, tömb és rekord. Tipus definiálás csak rekordformátum esetén lehetséges.

A számok lehetnek: byteinteger, shortinteger, integer, real és longreal, ezek rendre 1,2,4,4,8 byte-ot foglalnak. A számkonstansok lehetnek decimális, hexadecimális és bináris felirásuak. A string mindig hosszával megadott karaktersorozat és az XXPL-hez hasonlóan itt is alaptipusként kezelődik. Egy n hosszu string belső ábrázolása n+1 byte-ban történik, az első byte tartalmazza a string hosszát.

Példa string deklarálásra: %STRING(5) S,Z;

A tömb maximálisan 7 dimenziós lehet. A deklarációban alsó és felső indexhatárt kell megadni és ezek változók is lehetnek, tehát dinamikus tömbméret lehetséges.

Pl. %INTEGER ARRAY IN(1:K)

Hivatkozásban az index egész kifejezés lehet.

A rekord deklarálásához először a rekord strukturáját, a rekordformátumot kell definiálni. Ez tulajdonképpen a tipus definiálás csirája. A rekordformátum mezőtipus és mezőnév párokból álló lista. A rekord deklarálásakor egy már definiált rekordformátumra kell hivatkozni.

Pl. %RECORDFORMAT F (%INTEGER A,%STRING(8) S)
%RECORD R(F)

Egy rekord mezője szám, string, pointer, egydimenziós fixméretü tömb és rekord lehet. Rekord tipusu mezőnél a formátumot is meg kell adni és annak már definiáltnak kell lenni. Ez a kikötés kizárja a rekurziv rekordot. Ha a mező rekordra mutató pointer, akkor a formátumdefiniciót követő %RECORDSPEC utasitás specifikálja a hivatkozott rekord tipusát.

Példa:

```
%RECORDFORMAT F (%INTEGER I, %RECORDNAME J)
%RECORDFORMAT K (%REAL X,Y)
%RECORDSPEC   F_J(K)
```

A rekordokból álló tömb, azaz a %RECORDARRAY deklarálása a következőképpen történik.

Pl. %RECORDFORMAT F (%INTEGER A, %REALARRAY X(1:5))
%RECORDARRAY RA(1:100)(F)

A hivatkozás minősitett névvel történik értelemszerüen:

Pl. RA(100)_X(5)

A pointer változó tulajdonképpen nem változó csak név olyan értelemben, hogy nem foglalódik neki hely a memóriában, csak feljegyződik a szimbólum táblába.

Deklarációja is erre utal, a tipusnévhez NAME szót kell ragasztani.

Pl. %INTEGERNAME I,J,K

Felhasználása ott történik, ha ugyanarra a változóra különböző, de azonos tipusu nevekkel akarunk hivatkozni. Felhasználás

előtt mindig equivalenciába kell hozni egy deklarált változóval.

A deklarált változóknak lehet prefixük, ami a változó kezelésére vonatkozóan ad utasitást. Az IMP terminológia nem igy nevezi, de tulajdonképpen memória osztályokról van szó. A prefix lehet: %OWN, %CONST, %EXTERNAL, %ENTRINSIC.

OWN prefix esetén a rutinból való kilépés és visszatérés esetén visszakapja kilépéskori értékét (mint az ALGOL-60 own változója), ez felel meg a többi nyelv static memória osztályának.

A CONST kulcsszó konstansokat deklarál.

Az EXTERNAL kulcsszó a változót más rutinok számára is hozzáférhetővé, azaz globálissá teszi, ez a terminológia eltér a szokásostól.

OWN, CONST és EXTERNAL prefix esetén a változónak kezdőérték adható. Tömbök közül csak fix méretü egydimenziós tömbök kaphatnak kezdőértéket. A kezdőérték konstansok listája, ismétlási tényező megengedett. Rekord nem kaphat kezdőértéket. %ENTRINSIC prefix esetén a változó számára nem történik helyfoglalás, más rutinok %EXTERNAL változói között kell keresni.

A számokon a szokásos aritmetikai müveletek, bitenkénti logikai müveltek és shift müveletek hajthatók végre (az utóbbiak real-en nem). Aritmetikai müveleteknél tipus ellenőrzés van és automatikus szélesités (byte-ból integer, integerből real stb.), a csonkitást (realból integer) pedig utasitásra végzi (INT). A logikai müveleteknél 32 bitre egészit ki. Érdekes, hogy értékadás 4 féle lehetséges: =, ←, == és →.

= aritmetikai értékadás, tipus összeférhetetlenség esetén hibajelzés;

← értékadás csonkitással, ha a jobboldal hosszabb mint a baloldal, akkor letöri a felesleges részt;

== pointer változók értékadása, itt a jobboldal nem a baloldal értéke lesz, hanem helyette használható;

→ resolution string müvelet (ld. 3.2.6.5).

A string müveletek közül a konkatenáció és a resolution, a string alapfüggvények közül pedig a karakter kiemelés, hossz meghatározás és szelet kiemelés szerepel.

Az ugyanahhoz az adathoz való többszörös hozzáférésre egyik eszköz a pointer változó.

Pl. %INTEGERARRAY TABLE (1:10,1:10)
%INTEGERNAME BASE
BASE==TABLE(1,1)

Ezután TABLE(1,1) BASE néven hivatkozható.

Az un. "MAPPING" függvények segitségével ugyanazt az adatot különböző tipusuként is hivatkozhatjuk és összetett adatstrukturának más strukturát adhatunk.

Pl. %LONGREAL X
%INTEGERNAME I,J
I== INTEGER(ADDR(X))
J== INTEGER(ADDR(X)+4).

4.3.2.2 Az IMP vezérlési szerkezetei

A feltételes vezérlésátadásoknál feltételként reláció, vagy relációknak az AND és OR operátorral összekötött sorozata áll. A kiértékelés sorrendjét a zárójelezés dönti el.

Feltételes vezérlésátadás

Formája:

%IF F1 %THEN U1
%IF F1 %THEN U1 ELSE U2

Itt F1 feltétel, U1 és U2 pedig feltétel nélküli utasitás.

U1 lehet: - egyszerü feltétel nélküli utasitás (értékadás, rutin hivás, feltétel nélküli ugró utasitás)
- %START...%FINISH utasitás zárójelek közé zárt utasitás sorozat,
- %AND kulcsszóval összekapcsolt utasitás sorozat.

Példák:

%IF A>B %THEN PRINT (A,5,3)
%IF A=1 %OR B=1 %THEN A=3 %AND C=3
%IF A=0 %THEN A=1 %ELSE %START...%FINISH

Ha az %IF kulcsszót %UNLESS kulcsszóra cseréljük, akkor az utasitássorozat a feltétel hamis voltakor kerül végrehajtásra. Ha az utasitássorozat egyetlen utasitásból áll, akkor a feltétel elé kerülhet. pl. A=0 %IF A=B.

Ciklus utasitások

WHILE utasitás

Formája: %WHILE F1 %THEN U1

Itt F1 feltétel, U1 feltétel nélküli utasitás.

U1 lehet: - egyszerü feltétel nélküli utasitás,
- %AND kulcsszóval összekapcsolt utasitássorozat,
- %CYCLE...%REPEAT zárójelpár közé fogott utasitássorozat.

A fenti esetben a feltételvizsgálat a ciklustörzs lefutása előtt történik és a törzs lefutása a feltétel igaz voltakor következik be.

%UNTIL F1 %THEN U1

esetén először U1 végrehajtódik, majd F1 vizsgálata következik és F1 igaz voltakor ismétlődik U1 végrehajtása.

Ha U1 egyszerü feltétel nélküli utasitás, akkor előre hozható

U1 %WHILE F1.

Ennek értelmezése teljesen azonos az előzőkkel.

Megjegyezzük, hogy ez a terminológia ellentétes a megszokottal. UNTIL kulcsszó esetén általában a feltétel hamis voltára történik az ismétlés, a feltételvizsgálat és a ciklustörzs lefutásának sorrendjét pedig általában a felirás sorrendje szabja meg.

Ciklus utasitás ciklusváltozóval és szabályos ismétléssel

Formája: %CYCLE N1=K1,K2,K3 U1,U2... %REPEAT

Itt N1 integer változó, K1,K2,K3 a kezdőértéket, lépést és végértéket megadó egész kifejezések, Ui-k utasitások. A ciklusváltozó értékét a cikluson belül nem szabad változtatni.

A %CYCLE...%REPEAT ciklus alkalmazható feltétel vagy ciklusváltozó nélkül is. Ilyenkor a ciklusból való kiugrás %EXIT vagy feltétel nélküli vezérlésátadás segitségével történhet.

Esetszétválasztás:

Az esetszétválasztásra a switch szolgál, ezt 3.3.3-ban részletesen ismertettük, ezért itt nem térünk ki rá.

Feltétel nélküli vezérlésátadás

Formája: → cimke

A cimke azonositó vagy egész szám lehet (1-től 16383-ig) utána kettősponttal.

Pl. 27:

:

→ 27

Szökéskifejezés

Az %EXIT utasitás hatására a vezérlés elhagyja a ciklust és a %REPEAT utasitást követő utasitásra kerül, de a ciklus változó (ha van) pillanatnyi értéke megőrződik.

Rutin hivás és visszatérés

A rutin hivás alakja:

Nl(Pl,P2,...,Pn)

Itt Nl függvénynév, Pl,P2,...,Pn akutális paraméterek. Az érték szerint hivott paraméterek helyén kifejezés is állhat. Az aktuális paraméterek tipusának meg kell egyezni a formális paraméterek tipusával.

A rutinból való kilépés a %RETURN utasitásra vagy a rutin fizika végét is jelentő %END utasitásra történik.

Függvényeljárás esetén a függvényértéket meghatározó RESULT= <expr.> utasitás is a függvénytörzs elhagyását eredményezi.

4.3.2.3 Az IMP programszerkezete

Az összetett utasitások feltételes vezérlési szerkezetekben szerepelnek, mégpedig

%START ... %FINISH - IF szerkezetben

%CYCLE ... %REPEAT - WHILE szerkezetben

A változók scope-ját nem befolyásolják.

Blokk. Az ALGOL-60-ból jól ismert blokk struktura érvényes.

%BEGIN Dl,D2,...,Ul,U2,...%END

Itt Dl,D2,... deklarációk, Ul,U2,... utasitások.

A memória verem szervezésü, ez a blokk strukturát, a dinamikus tömbméretet és rekurziv függvényhivást is lehetővé teszi. A blokkok 10 mélységig lehetnek egymásba ágyazva. A blokk nyitott scope. A legkülső blokk, a főprogram megkülönböztetésül %ENDPROGRAM kulcsszóval végződik.

Eljárás és függvény

Az eljárásokat és függvényeket deklarálni kell. A deklaráció és specifikácóból és leirásból áll. A kettő elválhat egymástól. A specifikáció tartalmazza a nevet és a formális paraméterlistát. Formális paraméterlista a paraméterek tipusát és nevét tartalmazza. A leirás megismétli a specifikációt és leirja a törzset. A specifikációnak mindig meg kell előzni a hivást, a leirásnak nem.

Példa:

```
specifikáció: %ROUTINESPEC NAME(%REALNAME X, %INTEGER I)
leirás:       %ROUTINE NAME (%REALNAME X, %INTEGER I)
               deklarációk
               utasitások
               %RETURN
               utasitások
               %END
```

Függvénydeklarációban ROUTINE helyett <tipus> FN áll. Az eljárástörzs nyitott scope, a globális változók értékét az eljárás deklaráláskori környezete határozza meg és nem a hiváskori érték.

A paraméterátadás lehet érték szerinti és név szerinti. Érték szerinti paraméterátadás esetén a formális paraméter specifikációjában tipus és név szerepel, ilyen a fenti deklaráció 2. paramétere. Az aktuális paraméter ekkor a megfelelő tipusu kifejezés lehet.

Név szerinti paraméterátadásnál a tipusnévhez a NAME szó kapcsolódik, ezután következik a formális paraméter neve, mint a fenti deklaráció 1. paraméterénél. Ez esetben az aktuális paraméternek a megfelelő tipusu változónévnek kell lenni. Az ilyen paraméterre való hivatkozás a rutinon belül lényegében az aktuális paraméter által megnevezett nem lokális változóra való hivatkozás. A név szerint hivott paraméterek értéke a hivás pillanatában számitódik ki.

Tömböt és rekordot csak név szerint lehet átadni.

Megjegyezzük, hogy mivel az aktuális paraméterek értéke a hivás pillanatában számitódik ki, itt tulajdonképpen hivatkozás szerinti paraméterátadásról van szó, csupán az IMP terminológia

hivja ezt név szerinti paraméterátadásnak. Paraméterátadásnál szigoru tipusvizsgálat van. Eljárás és függvény is lehet paraméter. Az eljárások és függvények lehetnek rekurzivok.

Részenkénti fordithatóság

Egy IMP program részenként fordítható olymódon, hogy a külön leforditott részeket a könyvtárba visszük, futás közben az eljárások onnan hivják majd egymást.

Egy külön forditott részt az IMP file-nak nevez. Ez a következőképpen néz ki:

- globális változók %OWN vagy %CONST prefix-szel,
- rutin leirások %EXTERNAL prefix-szel,

%END OF FILE

Több file fordítható le ilymódon és ezeket a könyvtárba kell vinni.

A főprogram a következőképpen néz ki:

```
%BEGIN
%EXTERNALROUTINSPEC utasitással specifikáljuk a használni
                    kivánt rutinokat, azaz megadjuk a nevét és
                    a formális paraméterlistát.
%EXTRINSIC kulcsszó után felsoroljuk a másutt deklarált
                    globális változókat.
deklarációk
   :
utasitások
   :
%END OF PROGRAM
```

4.3.3 C-nyelv

A C-nyelvet Ausztriában Murray Hill-ben a Bell Telephone Laboratories-ban fejlesztették ki 1973-74-ben a korábban ugyanitt létrehozott B-nyelv továbbfejlesztéseképpen. Implementálva van a PDP-11 UNIX operációs rendszerében, a HIS 6070 computeren és az IBM 360/370 gépcsaládon. A nyelv jellemzője az operátorokban való gazdaság, a pointer tipusu változó nagyfoku kiépi-