Konteks bepaal betekenis

Byvoorbeeld, veranderlike `x` het verskillende tipes in prosedures `P` en `Q`:

```
PROCEDURE P;
  VAR x: INTEGER;
  PROCEDURE Q;
    VAR x: REAL;
  BEGIN
    x := 3.14
  END Q;
BEGIN
  x := 10
END P;
```

- Verklarings baken die bestek van veranderlikes af.
- Inligting vervat in verklarings moet gestoor word.

Simbooltabel

- Stoor inligting oor elke verklaarde simbool.
- Elke simbool wat teengekom word se inligting moet opgesoek word.
- Daar is verskillende soorte simbole (byvoorbeeld konstantes, veranderlikes en prosedurename).
- Daar is bestek ("scope") verbonde aan simbole—prosedures kan byvoorbeeld genes word.

Struktuur van simbooltabel

- Tabel is tradisioneel gebruik, maar dinamiese datastruktuur is meer gepas.
- Elke simbool het 'n inskrywing (rekord).
- Geskakelde lys van rekords werk goed, maar opsoek is stadig. Geskik vir tale waar min simbole per bestekvlak voorkom.
- Boom verbeter opsoekspoed, maar slegs indien gebalanseerd. Slegs nuttig indien baie simbole hanteer moet word.
- Hutstabel kan gebruik word, maar hantering van bestekvlakke is ingewikkeld.

Struktuur van inskrywings

Meeste inskrywings bevat die volgende velde:

- Die klas simbool (konstante, veranderlike, ens.)
- Die tipe van die simbool ('n veranderlike kan byvoorbeeld van tipe INTEGER of BOOLEAN wees)
- Die simbool se naam ('n string karakters)
- Die waarde, indien 'n konstante

Ontwerp van rekords vir inskrywings

Daar is twee gewilde opsies:

- Gebruik 'n basiese rekord met velde vir die naam van die simbool, 'n wyserveld na die volgende inskrywing, en die tipe van die simbool ('n wyser na 'n ander rekord wat die tipe beskryf)
- Gebruik 'n veld (van tipe INTEGER) om die klas van die simbool voor te stel. Simbole word dus geklassifiseer (konstantes, veranderlikes, tipes, prosedures) en elke klas se inskrywings het 'n spesifieke uitleg.

Voorstelling van tipes

- Tipes word voorgestel as rekords met die volgende velde:
  - Die vorm: basiese tipe (INTEGER, BOOLEAN) en gestruktureerde tipes (ARRAY of RECORD)
  - Verdere velde wat afhang van die vorm. Byvoorbeeld, 'n ARRAY het 'n lengte en element tipe. 'n Rekord het 'n lys velde, elk met 'n tipe.

Skep van nuwe inskrywings

- Gebruik 'n prosedure `NewEntity` om geheue toe te ken vir nuwe inskrywings.
- Stel vas of daar alreeds 'n entiteit met dieselfde naam in die simbooltabel is. Indien nie, ken geheue toe vir die inskrywing.
- Stuur 'n wyser terug na die inskrywingsrekord wat toegeken is.

Datavoorstelling tydens looptyd

- Geheue word gewoonlik adresseer as grepe, elk met 'n unieke adres.
- Veranderlikes wat agtereenvolgend verklaar is word gewoonlik toegeken met stygende of dalende adresse.
- Elke verwerker ondersteun sekere basiese tipes direk met groottes wat wissel van een vervaardiger na die volgende. Byvoorbeeld, wisselpunt getalle mag 4 grepe beslaan en vastepunt getalle 2 grepe.
- Adresse word toegeken as *verplasings* vanaf basis-adresse wat tydens looptyd bepaal word.

Hantering van tipe-verklarings

```
(* "TYPE" ident "=" type *)
IF sym = type THEN
  Get(sym);
  WHILE sym = ident DO
    NewEntity(ent, Typ); Get(sym);
    IF sym = eql THEN Get(sym)
    ELSE Error(1)
    END;
    Type(ent.type);
    IF sym = semicolon THEN Get(sym)
    ELSE Error(2)
    END
  END
END;
```


Hantering van veranderlike-verklarings

```
(* "VAR" ident {"'" ident} ":" type *)
IF sym = var THEN
  Get(sym);
  WHILE sym = ident DO
    IdentList(Var, first);
    Type(tp);
    ent := first; (* start at first one *)
    WHILE ent # last entity in list DO
      ent.type := tp; INC(adr, ent.type.size);
      ent.val := -adr; ent := ent.next
    END;
    IF sym = semicolon THEN Get(sym)
    ELSE Error(2)
    END
  END
END
```


Lyste veranderlikes

- Ken 'n inskrywingsrekord toe vir elke veranderlike in die lys.
- Probleem: in Oberon-0 word die tipe van veranderlikes in 'n lys eers teegekom aan die einde van die lys.
- Oplossing: merk die eerste veranderlike in die lys sodat die tipe later ingevul kan word.
- Alternatiewe oplossing: gebruik 'n rekursiewe prosedure om 'n lys veranderlikes te verwerk.

Verwerking van Tipes

- Basiese tipes en verklaarde tipes is inskrywings in simbooltabel.
- Soek inskrywing op wat tipe beskryf (wyser word teruggestuur).
- Gebruik wyser om die inskrywing van die veranderlike te voltooi.

Skikkings

- Herken sleutelwoord `ARRAY`.
- Grootte volg as 'n positiewe heelgetal.
- Herken sleutelwoord `OF`.
- Verkry wyser na inskrywing vir element tipe van skikking.
- Ken geheue toe vir inskrywing vir skikking en vul velde in.

Rekords

- Herken sleutelwoord `RECORD`.
- Ken geheue toe vir inskrywing.
- Betree nuwe vlak (rekord velde is een vlak dieper genes).
- Hanteer lys veranderlikes en hulle tipes en vul alle velde in.
- Val terug na vorige vlak aan die einde van rekord se verklaring.
- Herken sleutelwoord `END` wat verklaring van rekord sluit.

Gebruik van inligting in simbooltabel

- Opspoor van simbole wat meer as een keer verklaar word (fout).
- Om te bepaal of die tipe van 'n uitdrukking versoenbaar is met die tipe van die veranderlike waaraan dit toegeken word.
- Om detail omtrent veranderlikes (byvoorbeeld die adres) te verkry tydens kodegenerasie.

Implementering in C

Verklaar inskrywings in simbooltabel so:

```
typedef struct EntityDesc *Entity;

struct EntityDesc {
  int class;
  Entity next;
  /* other fields */
};
```

Om 'n inskrywing dinamies toe te ken en 'n wyser `p` daarna te verkry:

```
p = malloc(sizeof(struct EntityDesc));
```

Om te verwys na 'n veld in 'n inskrywing via wyser `p`:

```
p->class = Array; /* Array is constant 1 */
```