Metody přidělování paměti

Základní způsoby: -Statické (přidělění paměti v čase překladu)
-Dynamické (přiděleno v run time) — **v zásobníku**, **na haldě**

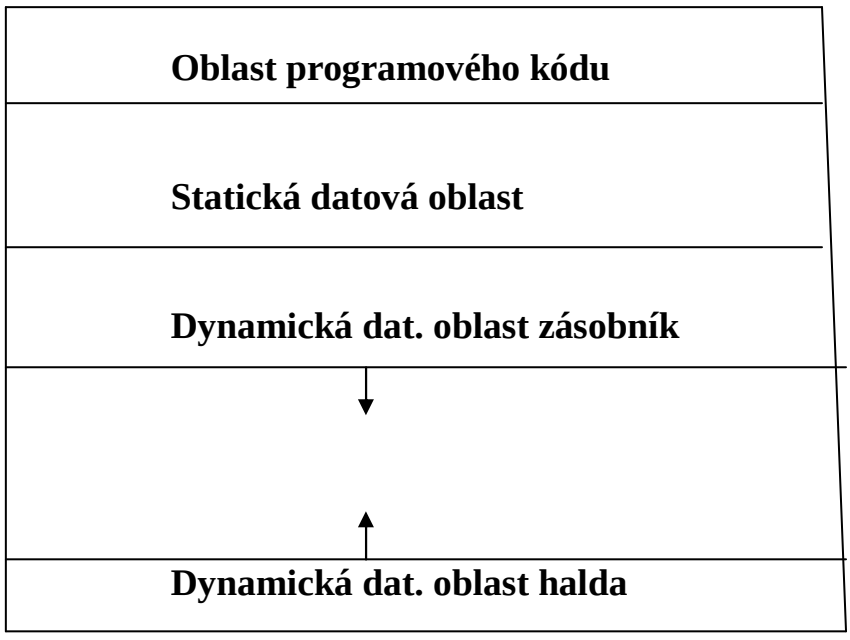
Důležitá hlediska jazykových konstrukcí:

- **Dynamické typy**
- **Dynamické proměnné**
- **Rekurze**
- **Konstrukce pro paralelní výpočty**

Podstatný je rovněž způsob:

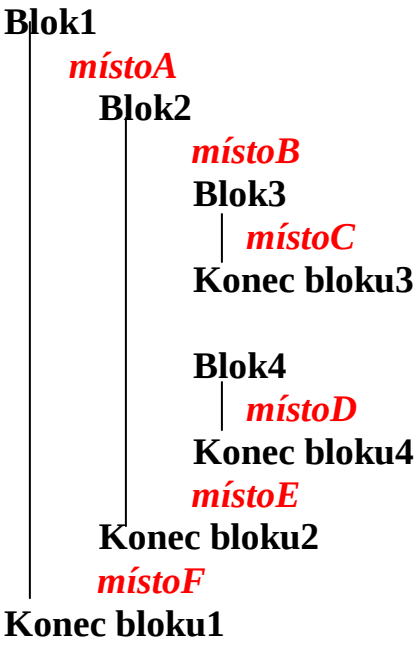
- **Omezování existence entit v programu (namespace, package, blok…)**
- **Určování přístupu k nelokálním entitám**
  na základě statického vnořování rozsahových jednotek,
  na základě dynamického vnoření rozsahových jednotek.

Rozdělění paměti cílového programu



Statické přidělování paměti

- **Globální proměnné**
- **Static proměnné**
- **Proměnné jazyka bez rekurze (i s blokovou strukturou)**



Statické přidělování pomocí zásobníku

? obsah zásobníku v různých okamžicích výpočtu

Bloky

		3	4		
	2	2	2	2	
1	1	1	1	1	1
Místo A	**B**	**C**	**D**	**E**	**F**

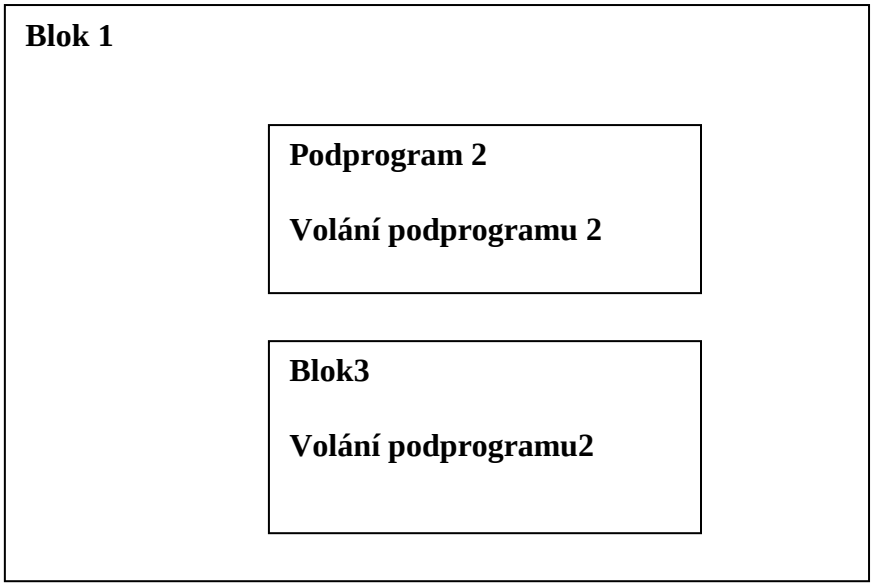
Dynamické přidělování v zásobníku

Aktivační Záznam (AZ představuje lokální prostředí výpočtu)

obsahuje místo pro:

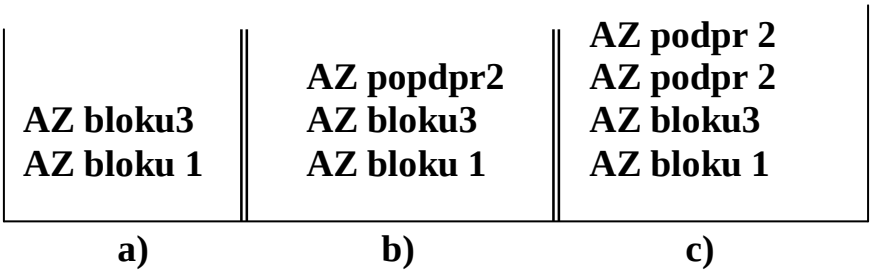
- **Lokální proměnné**
- **Parametry**
- **Návratovou adresu**
- **Funkční hodnotu (je-li podpr. funkcí)**
- **Pomocné proměnné pro mezivýsledky (také možno v registrech)**
- **Další informace potřebné k uspořádání aktivačních záznamů**

Př.

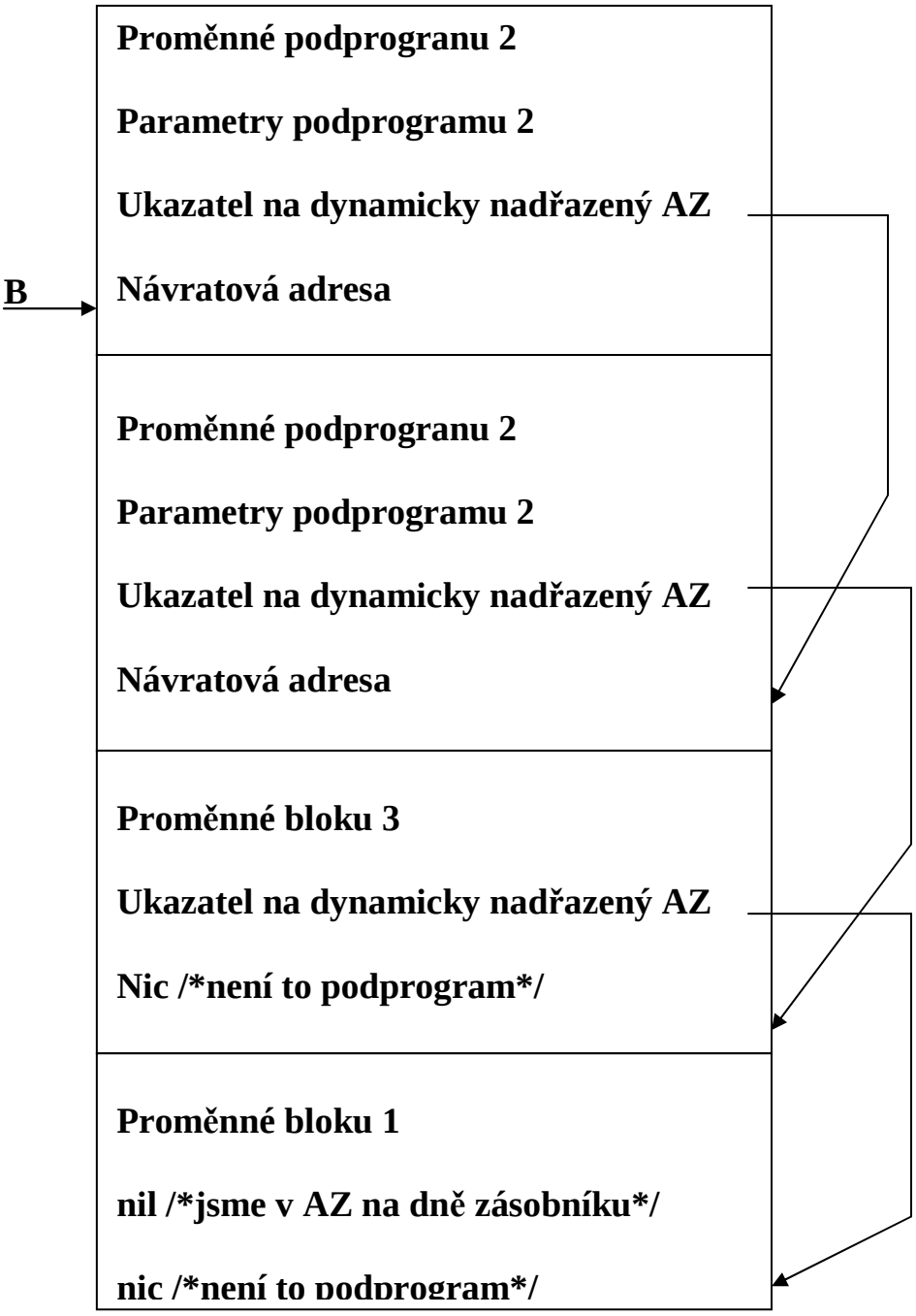


? stav výpočtového zásobníku

a) **Při vstupu do bloku 3**
b) **Při prvním volání podprogramu 2**
c) **Při rekurzivním vyvolání podprogramu 2**



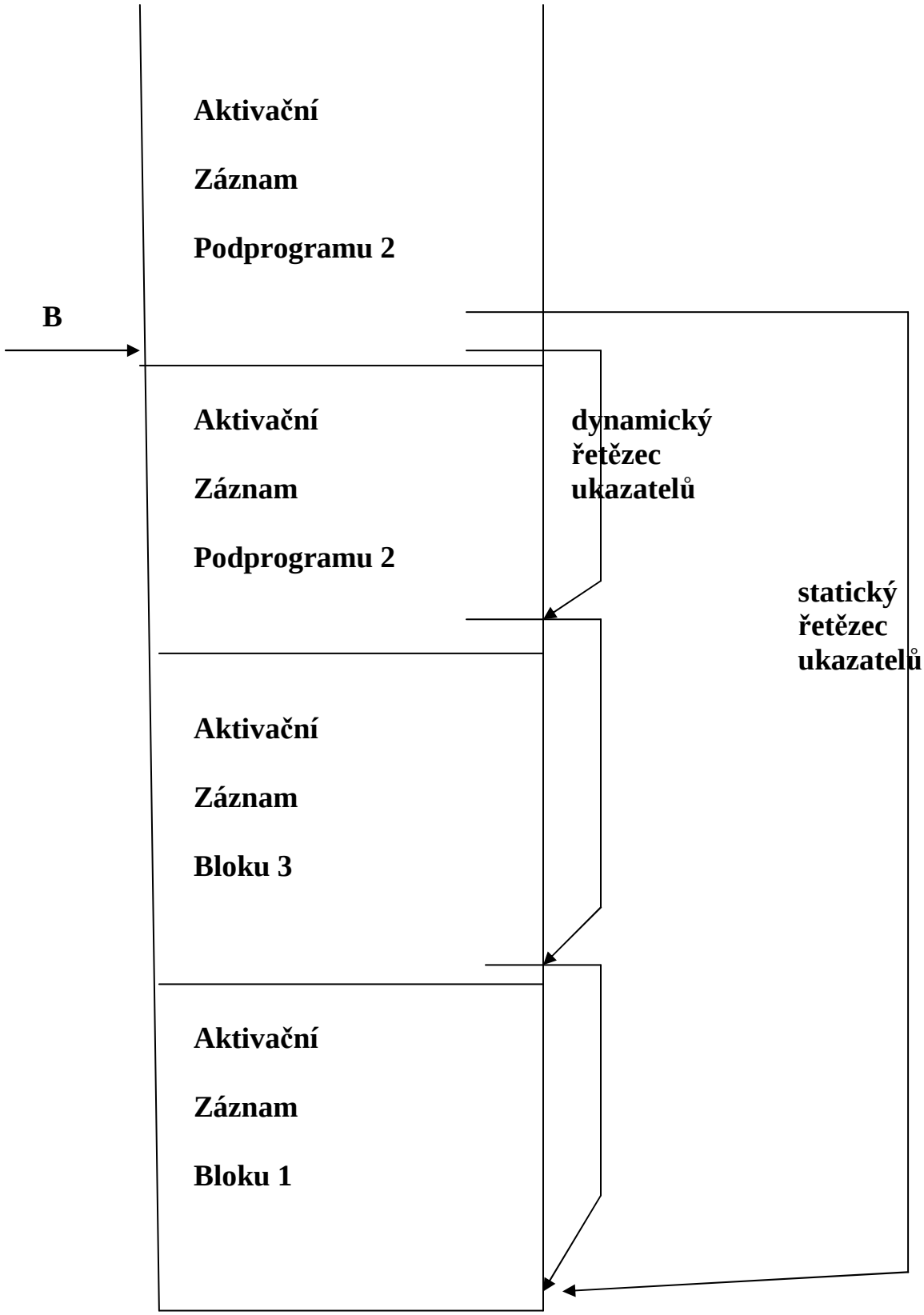
Uspořádání aktivačních záznamů při rekurz. vyvolání podpr.2
s dynamickým řetězcem pro rušení AZ při výstupu z rozsahové jednotky



Obr. Zásobník při rekurzivním volání podprogramu 2

B je registr ukazující na vrcholový AZ

Potřebujeme ještě vyřešit přístup k nelokálním proměnným při statickém = lexikálním rozsahu platnosti jmen. To řeší tzv. řetězec statických ukazatelů



Obr. Zásobník se statickým a dynamickým řetězcem ukazatelů při rekurzivním volání podprogramu 2

Vytváření řetězců ukazatelů

Nechť AZ má tvar: **pomocné proměnné**
Lokální proměnné
Parametry
Funkční hodnota
Statický ukazatel
Dynamický ukazatel
Návratová adresa

Zásobník Z, s vrcholem T

Při stupu do rozsahové jednotky (vyvolání podprogramu nebo vstupu výpočtu do bloku = Aktivace rozsahové jednotky):

A1) Z[T + 1] ← návratová adresa /* pouze u podprogramů*/
A2) Z[T + 2] ← B /*nastavení dynamického ukazatele*/
A3) Z[T + 3] ← B
For i ← 1 to m – n do Z[T + 3] ← Z[Z[T + 3] + 2] /*nastavení statického ukazatele*/
A4) B ← T + 1 /*nastavení bázového registru*/
A5) T ← T + velikost aktivačního záznamu
A6) skok na první instrukci podprogramu a uložení do Z údajů o skutečných parametrech /*pouze u podprogramů*/

Pozn. Je-li podprogram překládán odděleně (neznámá velikost jeho AZ), pak je úprava T provedena až na začátku volaného podprogramu.

Při výstupu z rozsahové jednotky (Návrat z podprogramu nebo průchod koncem bloku):

N1) T ← B – 1
N2) B ← Z[B + 1]
N3) skok na adresu uloženou v Z[T + 1] /*pouze u podprogramů*/

Výstup z rozsahové jednotky nelokálním Skokem (hladina n deklarace návěští je menší než hladina m místa s příkazem skoku)
Vždy platí n ≤ m

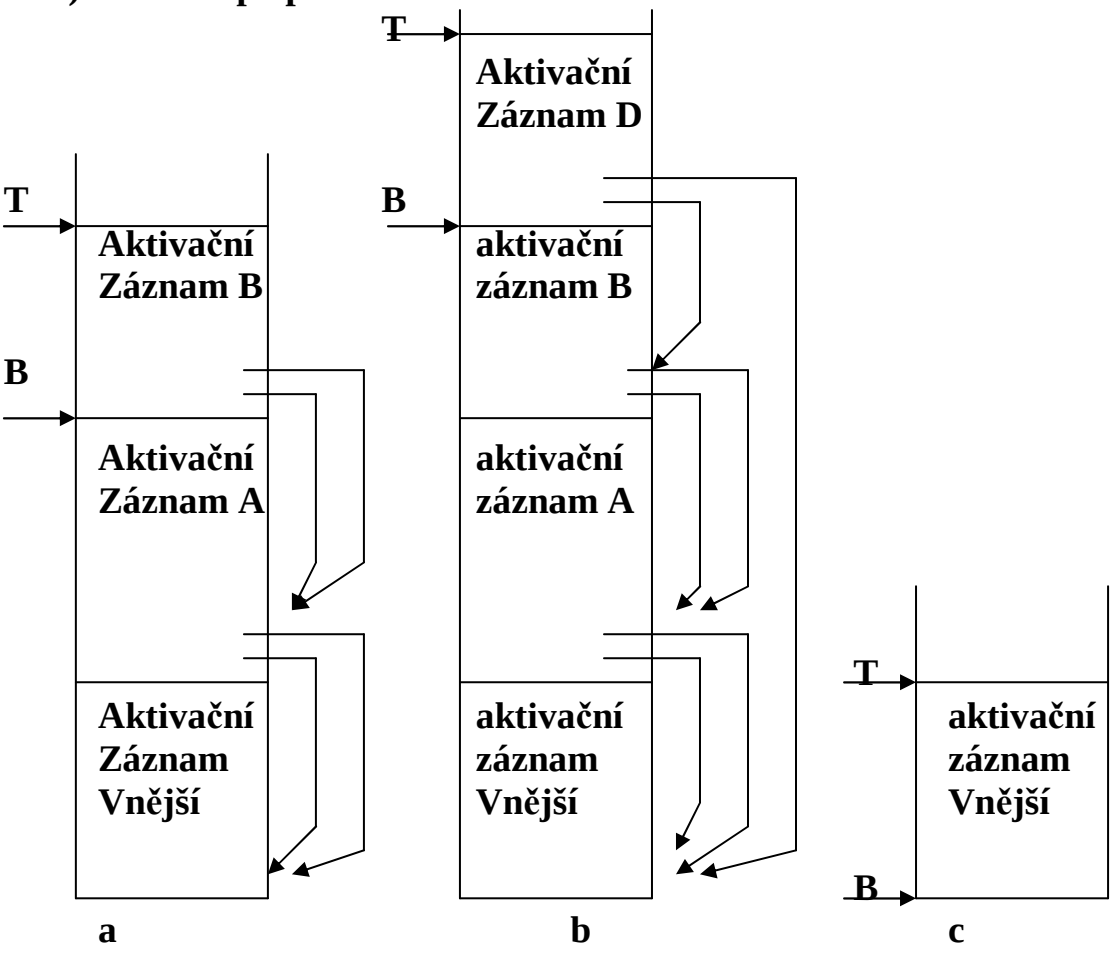
```
S1) for i ← 1 to m – n do { Pom ← B
                            repeat   T ← B – 1
                                     B ← Z[ B + 1]
                            until B ≠ Z[ POM + 2]
                          }
```

S2) skok na adresu, kterou návěští představuje

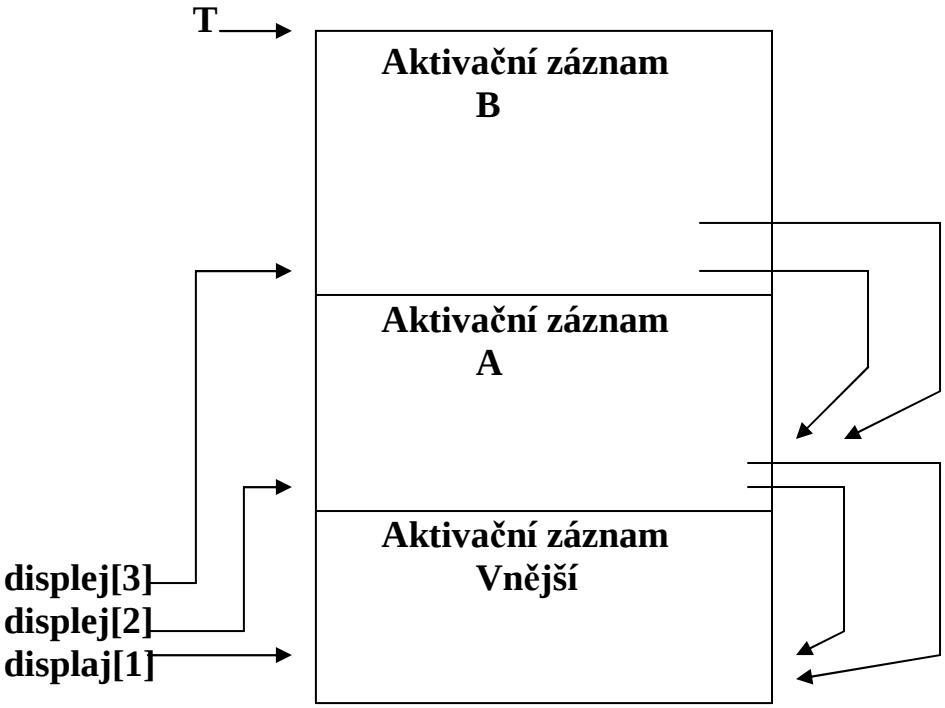
Př.

```
Podprogram Vnější
Label 1
        Podprogram A
                Podprogram B
                Volání D
        Volání B
        Podprogram D
        Goto 1
Volání A
1:
```

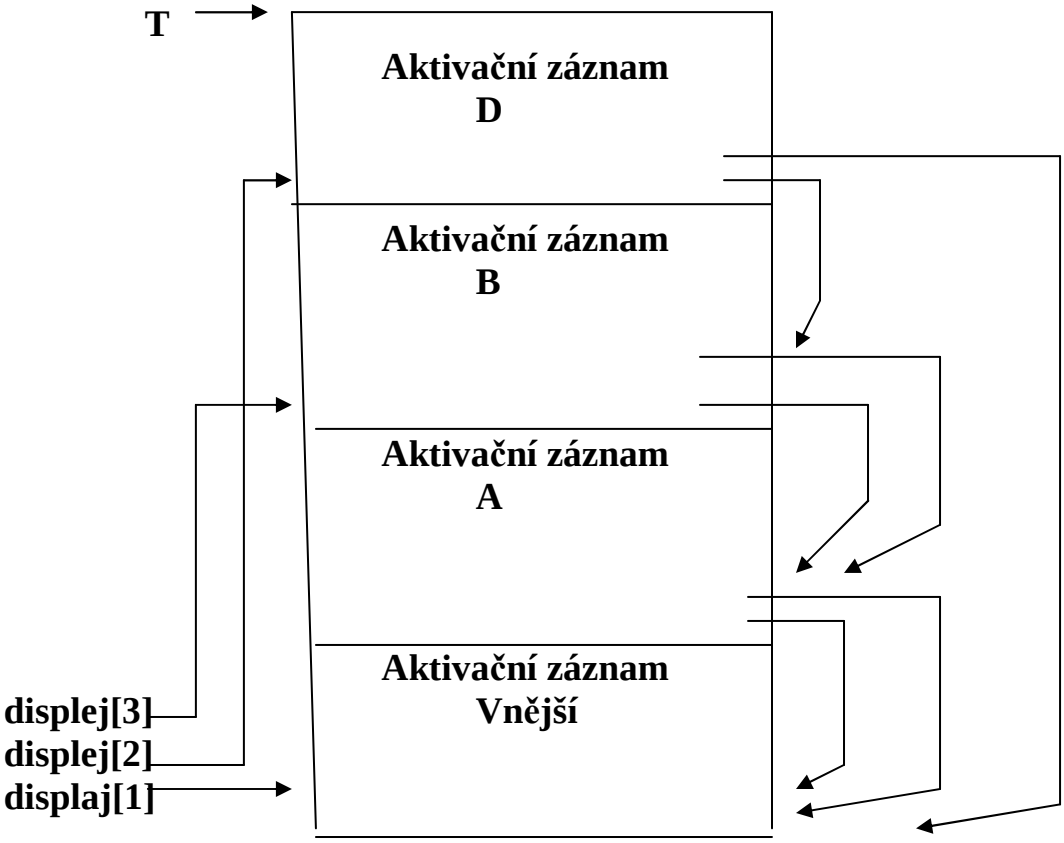
a) **obsah Z při provádění B, před voláním D,**
b) **obsah Z po vyvolání D, ped provedením nelokálního skoku,**
c) **obsah Z po provedení skoku.**



Zrychlení přístupu k nelokálním proměnným
(pomocí vektoru ukazatelů displej[i], kde i je hladina rozs. jedn.)



Obr. Stav Z při výpočtu B

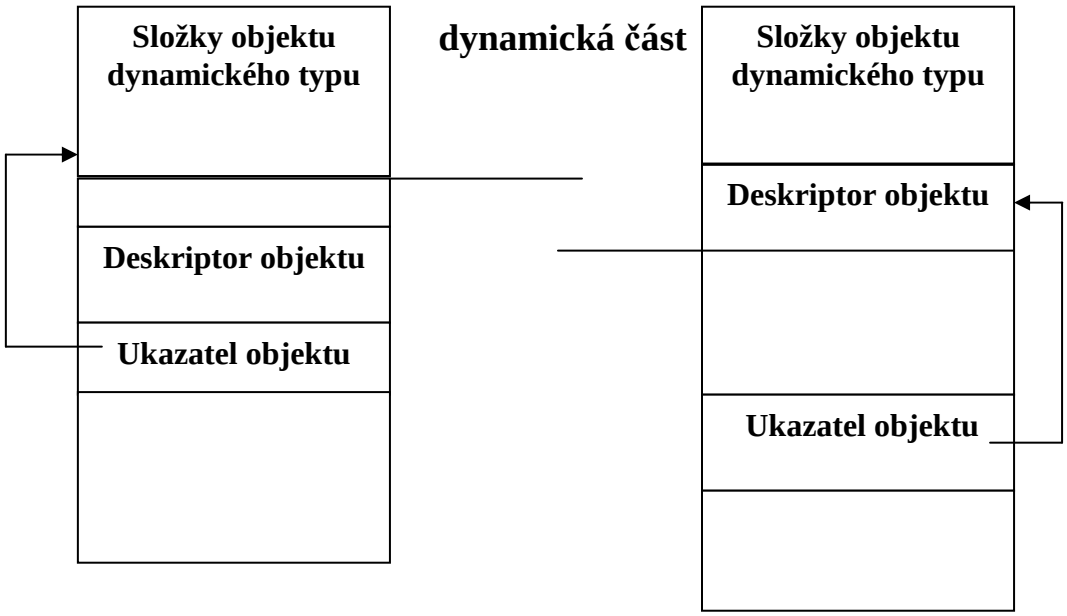


Obr. Stav Z při výpočtu D

Dynamická adresa (n, p) = displej[n] + p

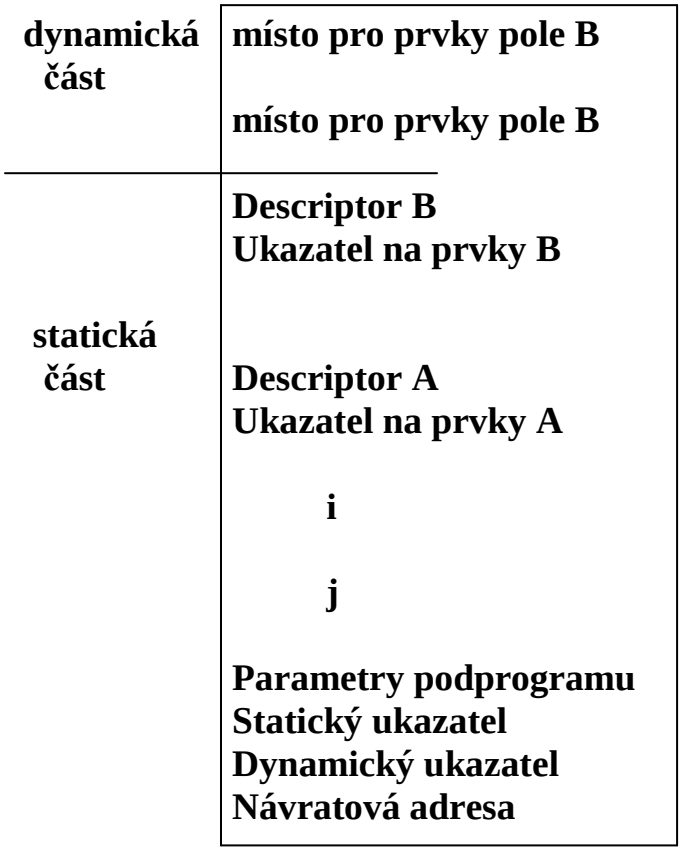
Objekty s dynamickými typy

Možnosti struktury aktivačního záznamu s objektem dynamického typu



Př. Aktivačního záznamu s objekty dynamického typu

```
podprogram PRIKLAD;
        int i, j ;
        int A(m .. n);
        int B(p .. q, r .. s,  );
```



Předávání parametrů podprogramům

-hodnotou (C, C++, Java,C#) formální parametr je lokální proměnnou do níž se předá hodnota

-odkazem (C, C++ je-li parametrem pointer, objektové parametry Javy, C# označené ref) předá informaci o umístění skutečného parametru

-výsledkem - formální parametr je lokální proměnnou z níž se předá hodnota do skutečného parametru před návratem z podprogramu

-hodnotou výsledkem (novější Fortran) - kombinace

-jménem – má efekt textové substituce (jako historická zajímavost)

-strukturované parametry

-statické typy ⇒ adresa prvého prvku

-dynamické typy ⇒ ukazatel na descriptor

-podprogramy

-u jazyků nedovolujících hnízdění podprogramů ⇒

adresa začátku = pointer

- u jazyků dovolujících hnízdění podprogramů ⇒

Spolu s adresou musí předdat i platné prostředí

-mělká vazba ⇒ platné je prostředí v němž se nachází volání formálního podprogramu

-hluboká vazba ⇒platné je prostředí kde je předávaný podprogram definován

-ad hoc vazba ⇒platné je prostředí kde je vydán příkaz volání podprogramu jež má za parametr podprogram

Př.

```
Podprogram  P1( )  {
        Prom x ;
        Podprogram P2 ( )  {
                Vytiskni (x) ; /*co se tady tiskne?*/
        };
        Podprogram P3 ( )  {
                Prom x ;
                x ← 3;
                P4(P2) ;
        };
        Podprgram P4( podprogram Px )  {
                Prom x ;
                x ← 4;
                call Px( );
        }
        x←1;
        P3( ) ;
}
```

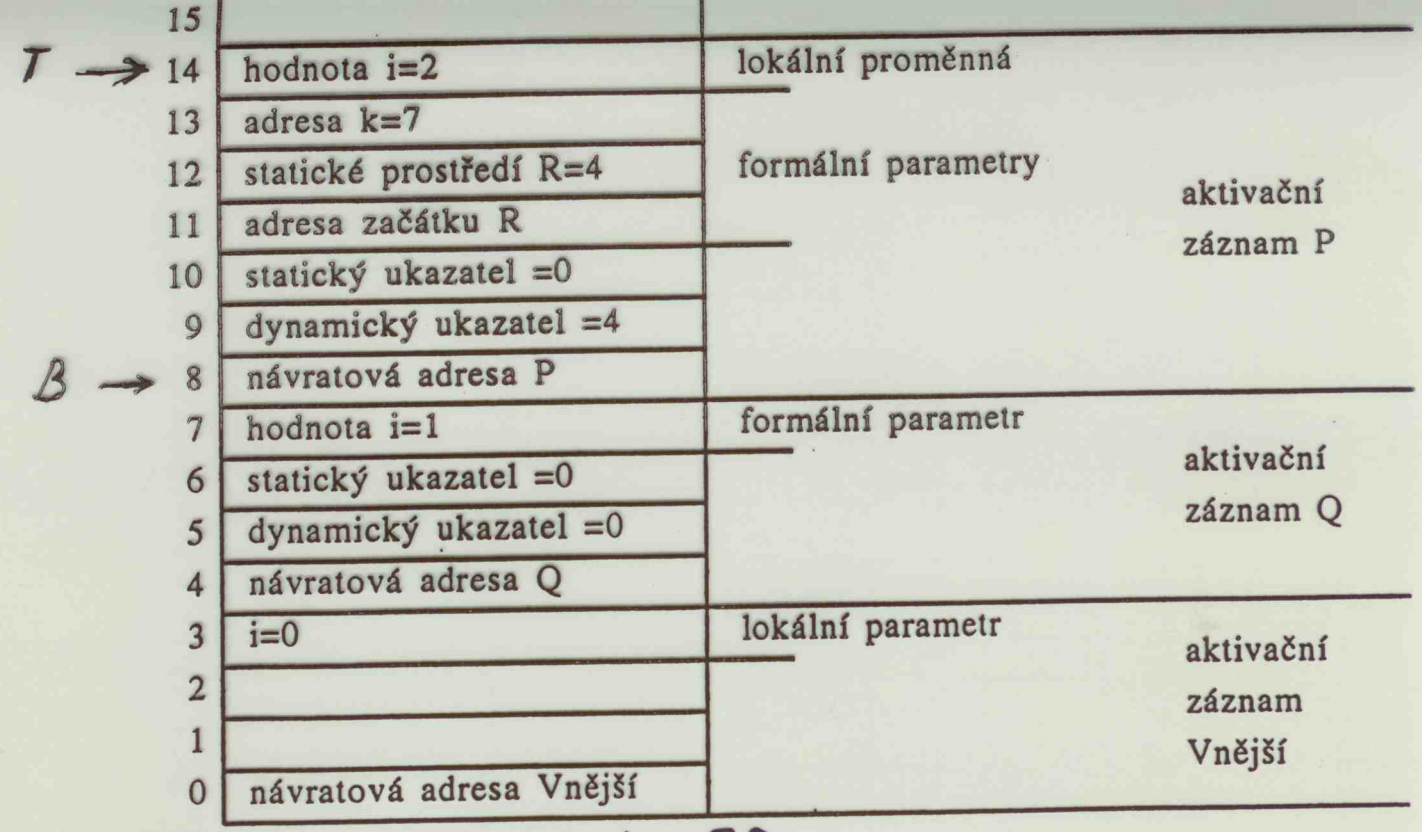
Při mělké vazbě se tiskne …

Při hluboké vazbě se tiskne …

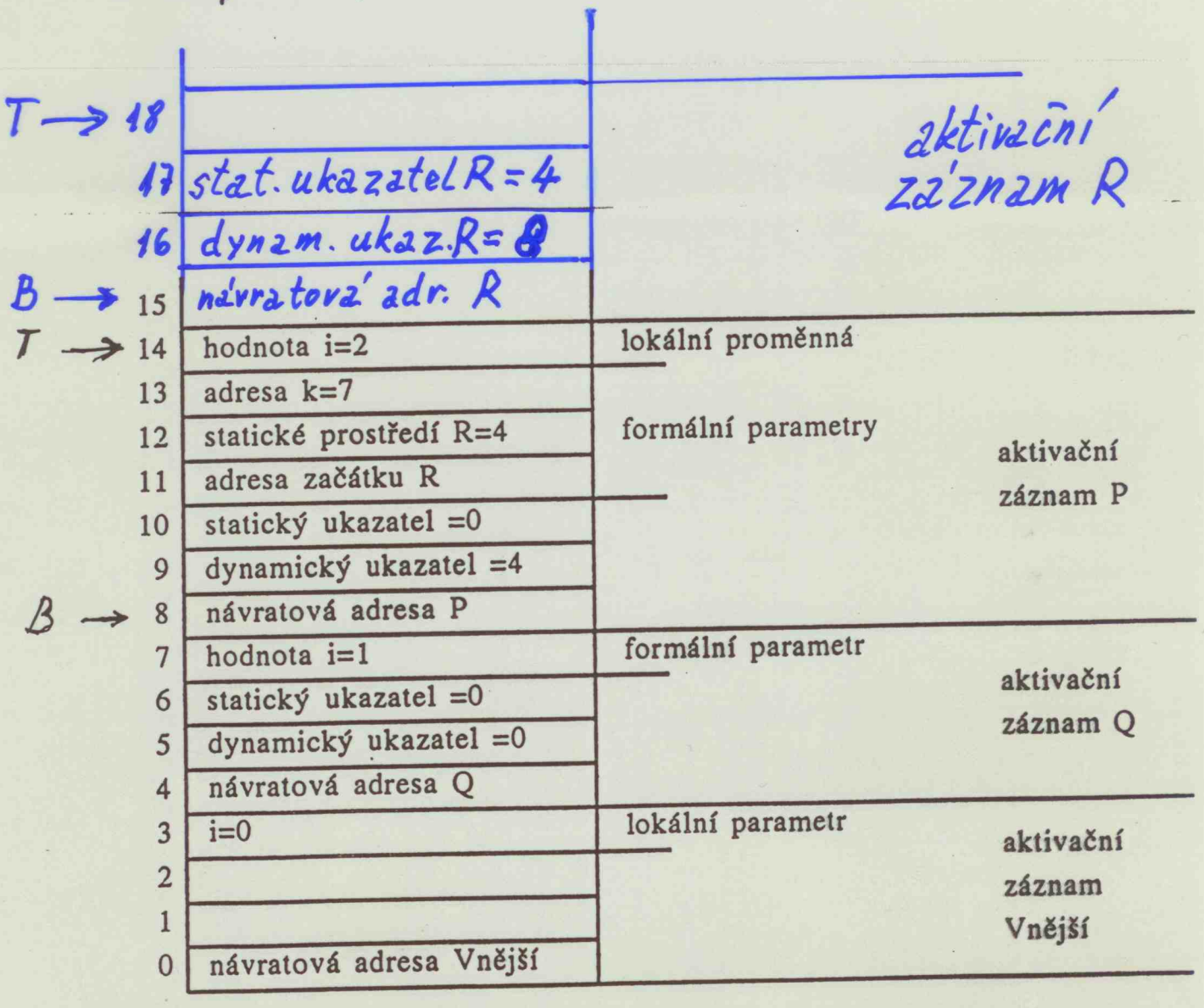
Při ad hoc vazbě se tiskne …

Př.
Předpokládejme hlubokou vazbu. Co se vytiskne po spuštění procedury Vnější?

```
podprogram Vnejsi;  {
        prom i:int;
        podprogram P( podprogram FP; prom k:int;)  {
                prom i:int;
                i←k+1;  FP( ); tisk(i);
        }
        podprogram Q(i:int);
                podprogram R ( )  {
                        Tisk(i);
                }
                P(R,i);
        }
        i← 0; Q(i+1);
}
```



Stav před vyvoláním FP



Stav po vyvolání FP

Paralelní zásobník

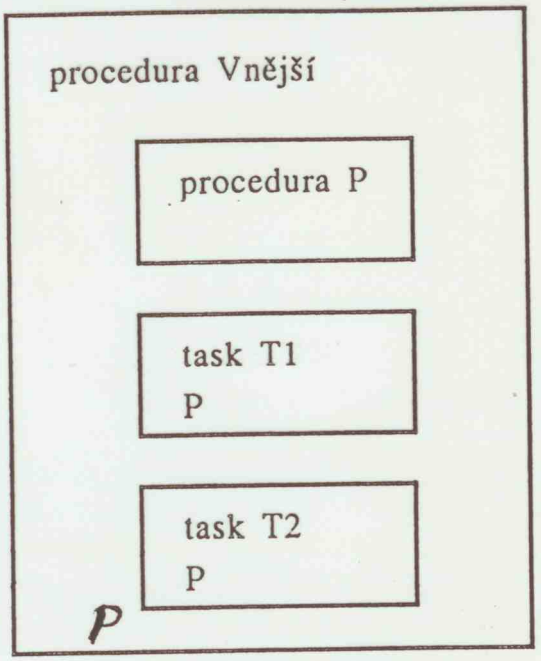
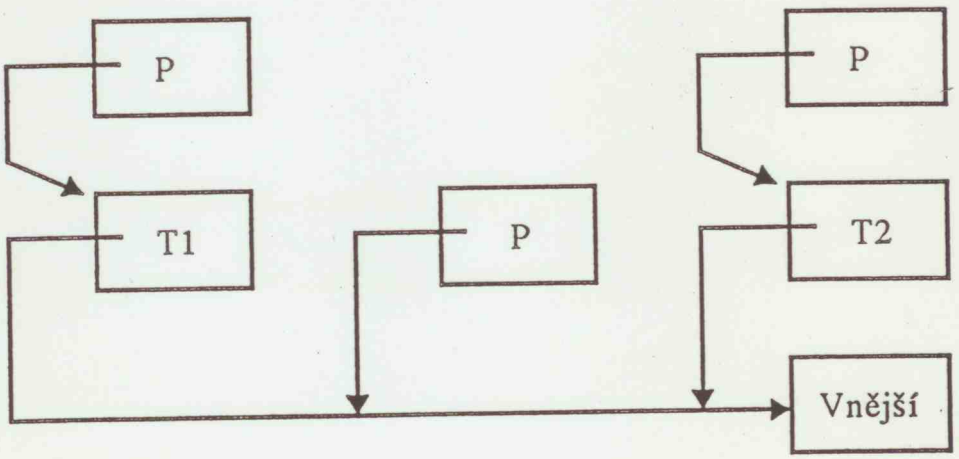
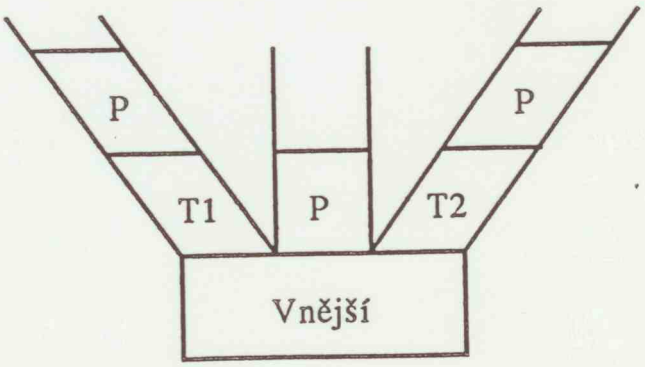


Schéma vnoření bloků programu



Struktura aktivačních záznamů při paralelním výpočtu



Uložení aktivačních záznamů ve zobecněném zásobníku