Reeglid

Reeglile vastav kood:

- reserveerib magasinis mälu lokaalsetele muutjuatele;
- väärtustab reegli keha;
- vabastab magasini freimi (kui võimalik).

Reeglis sisalduvaid lokaalseid muutujaid tähistame $\{X_1, \ldots, X_m\}$.

NB! Esimesed k lokaalset muutujat on formaalsed parameetrid.

$\text{code}_C\ (p(X_1, \ldots, X_k) \leftarrow g_1, \ldots, g_n)\ \rho$ = `pushenv m`

$\text{code}_G\ g_1\ \rho$

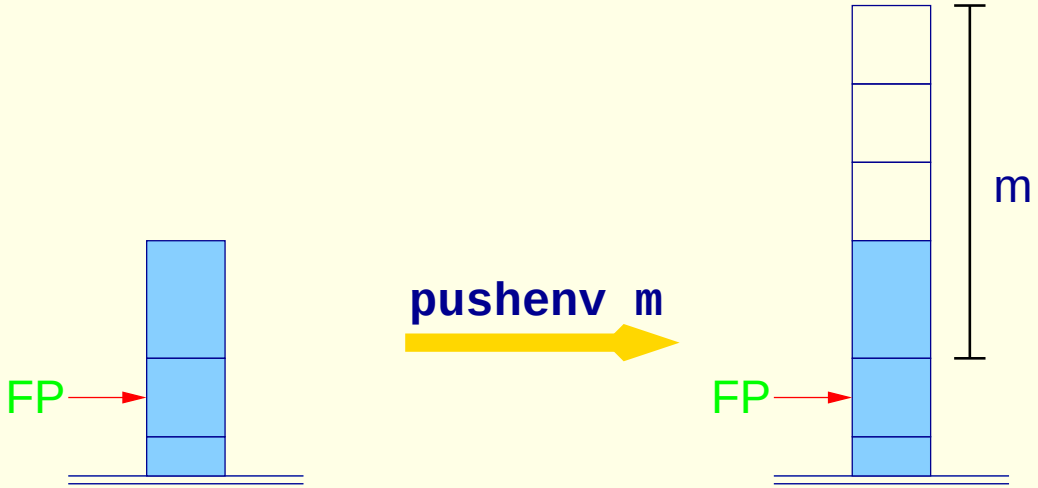
$\ldots$

$\text{code}_G\ g_n\ \rho$

`popenv`

Reeglid

Käsk `pushenv m` reserveerib mälu lokaalsetele muutujatele



SP = FP + m;

Reeglid

Näide: olgu antud reegel

$$r \equiv \mathtt{a}(X,Y) \leftarrow \mathtt{f}(\overline{X},X_1),\ \mathtt{a}(\overline{X_1},\overline{Y})$$

Siis $\mathrm{code}_C\ r$ emiteerib koodi:

```
    pushenv 3          call f/2          putref 2
    mark A          A: mark B            call a/2
    putref 1           putref 3       B: popenv
    putvar 3
```

Predikaadid

- Predikaat `q/k` on defineeritud reeglite jadana $rr \equiv r_1 \ldots r_f$.
- Predikaatide transleerimine toimub funktsiooni code_P abil.
- Juhul, kui predikaat on defineeritud ühe reegliga (so. $f = 1$), siis:

$$\text{code}_P\ r \quad = \quad \text{code}_C\ r$$

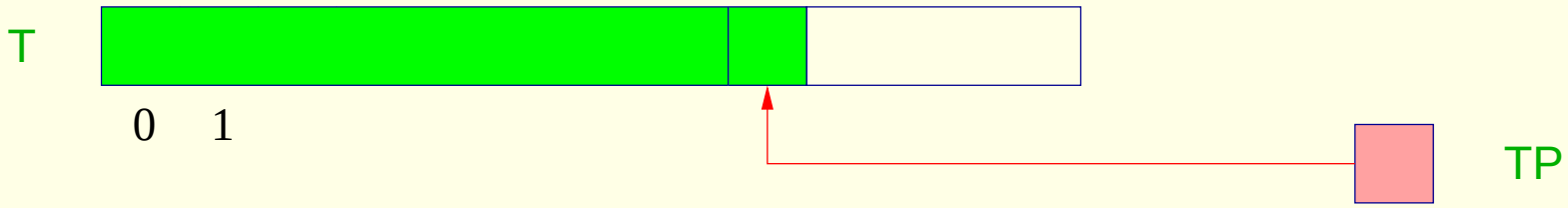
- Kui predikaat on defineeritud mitme reegli abil, siis:
  - tuleb ”proovida” esimest reeglit;
  - kui see ebaõnnestub, tuleb proovida järgmist reeglit; jne.

Predikaadid

- Kui unifitseerimine ebaõnnestub, kutsutakse välja täitmisaegne funktsioon `backtrack()`.
- Eesmärk on kerida kogu arvutuskäik tagasi kuni nn. *tagasipöördumispunktini* (backtrack point); so. (dünaamiliselt) viimase eesmärgini, kus saab järgmist reeglit ”proovida”.
- Selleks, et taastada varem kehtinud muutujate seosed, kasutati uute seoste loomiseks funktsiooni `trail()`.

Predikaadid

Jälg:

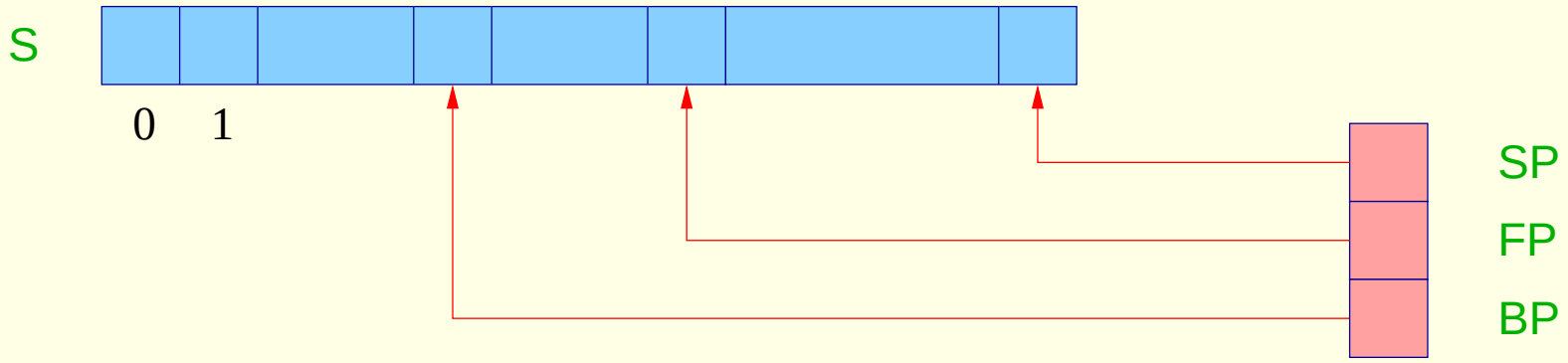


T = Trail — mälupiirkond loodud seoste hoidmiseks;

TP = Tail-Pointer — viitab viimasele kasutatud pesale.

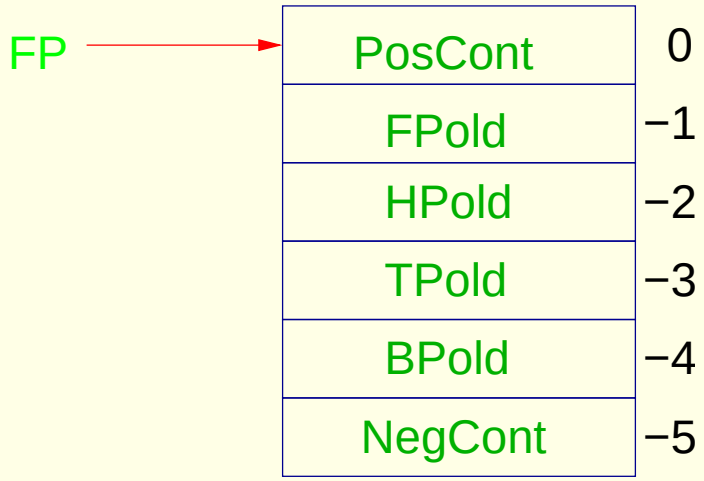
Predikaadid

Hetkel kehtivale tagasipöördumispunktile viitab register BP.



Predikaadid

Tagasipöördumispunkti esitab magasinis freim:



Parema loetavuse huvides kasutame edaspidi makrosid:

`PosCont`	≡	`S[FP]`	`TPold`	≡	`S[FP−3]`
`FPold`	≡	`S[FP−1]`	`BPold`	≡	`S[FP−4]`
`HPold`	≡	`S[FP−2]`	`NegCont`	≡	`S[FP−5]`

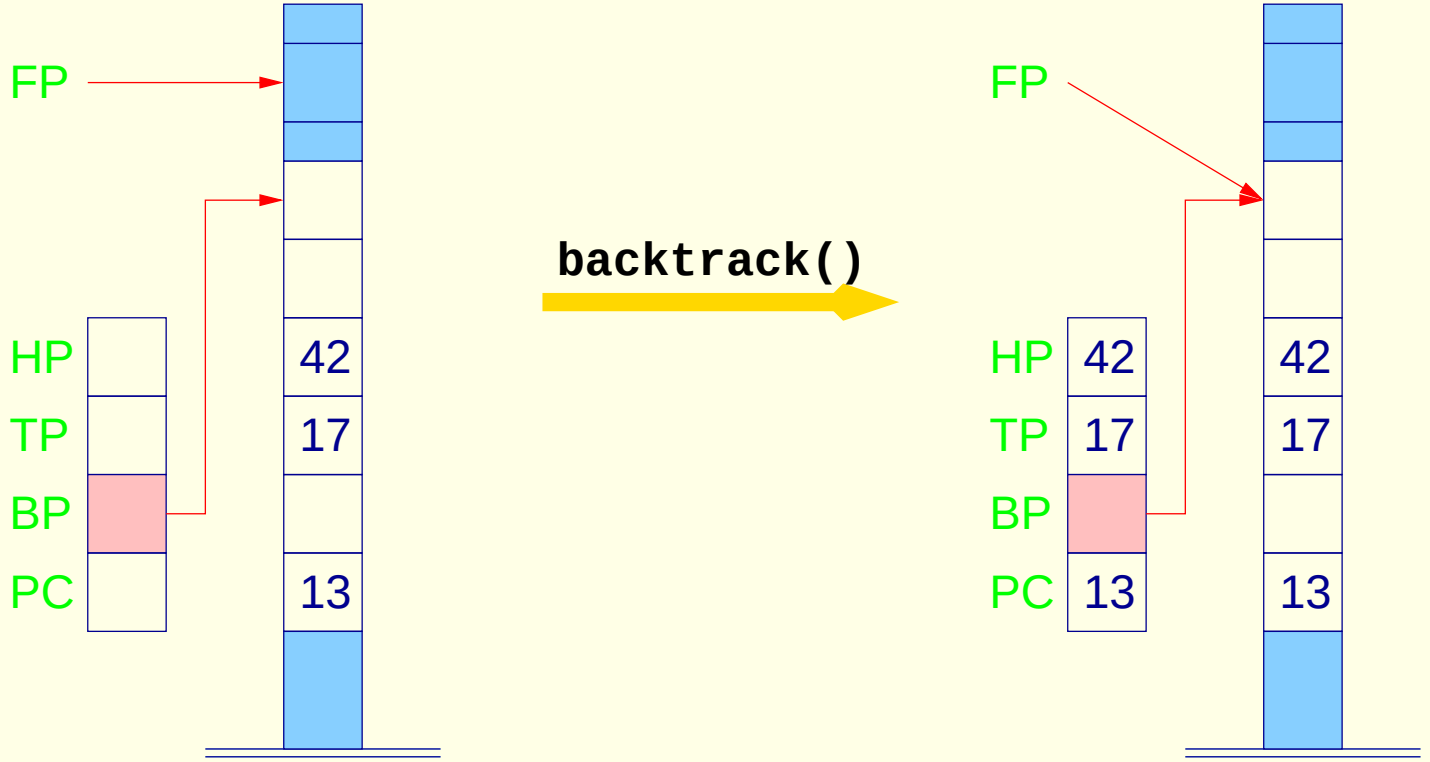
Predikaadid

Täitmisaegne funktsioon `backtrack()` taastab registrite seisu vastavalt tagasipöösrdumispunkti freimile:

```
void backtrack() {
    FP = BP;
    HP = HPold;
    reset (TPold,TP);
    TP = TPold;
    PC = NegCont;
}
```

Funktsioon `reset()` taastab muutujate seosed.

Predikaadid



Predikaadid

- Pärast viimast tagasipöördumispunkti loodud muutujad ja seosed saab eemaldada lihtsalt taastades registri HP vana väärtuse.
- See töötab juhul kui *nooremad* muutujad alati viitavad *vanematele* objektidele.
- Seosed, kus *vanemad* muutujad viitavad *noorematele* objektidele, tuleb eemaldada ”käsitsi”.
- Need seosed on salvestatud ”jäljes” (trail).

Predikaadid

Funktsioon `trail()` lisab uue seose juhul kui argument viitab nooremale objektile:

```
void trail (ref u) {
  if (u < S[BP-2]) {
    TP = TP+1;
    T[TP] = u;
  }
}
```

Pesa `S[BP-2]` sisaldab registrit HP enne tagasipöördumispunkti loomist.

Funktsioon `reset()` eemaldab kõik pärast viimast tagasipöördumispunkti lisatud seosed:

```
void reset (ref x, ref y) {
  for (ref u=y; x<u; u--)
    H[T[u]] = (R,T[u]);
}
```

Predikaadid

Predikaadi `q/k`, mis on defineeritud reeglitega $r_1, \ldots, r_f \quad (f > 1)$ transleerimisel genereeritakse kood, mis:

- loob tagasipöördumispunkti;
- üksteise järel "proovib" erinevaid reegleid;
- kustutab tagasipöördumispunkti.

Predikaadid

$$\begin{array}{lll} \text{code}_P\ (r_1,\ldots,r_f) & = & \text{q/k: } \texttt{setbtp} \qquad\qquad\quad \texttt{jump } \mathrm{A}_f \\ & & \qquad\quad \texttt{try } \mathrm{A}_1 \qquad\quad \mathrm{A}_1\text{: } \text{code}_C\ r_1 \\ & & \qquad\quad \ldots \qquad\qquad\quad \ldots \\ & & \qquad\quad \texttt{try } \mathrm{A}_{f-1} \qquad \mathrm{A}_f\text{: } \text{code}_C\ r_f \\ & & \qquad\quad \texttt{delbtp} \end{array}$$

NB!

- Tagasipöördumispunkt kustutatakse enne viimase reegli ”proovimist”.
- Viimasele reeglile hüpatakse ja kehtivale freimile enam tagasi ei pöörduta.

Predikaadid

Näide:

$$\begin{aligned} \mathtt{s}(X) &\leftarrow \mathtt{t}(\overline{X}) \\ \mathtt{s}(X) &\leftarrow \overline{X} = a \end{aligned}$$

Predikaadi $\mathtt{s}/1$ transleerimisel emiteeritakse kood:

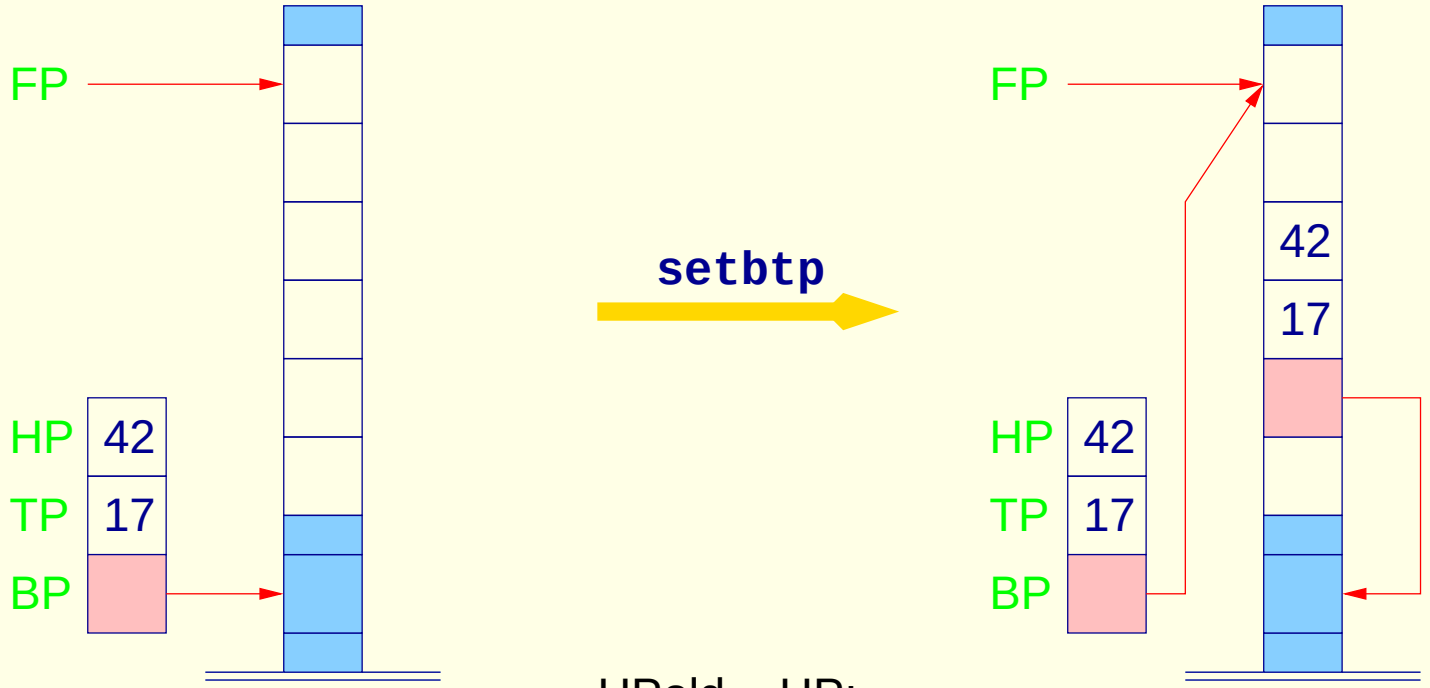
```
s/1: setbtp
     try A
     delbtp
     jump B

A:   pushenv 1
     mark C
     putref 1
     call t/1
C:   popenv

B:   pushenv 1
     putref 1
     uatom a
     popenv
```

Predikaadid

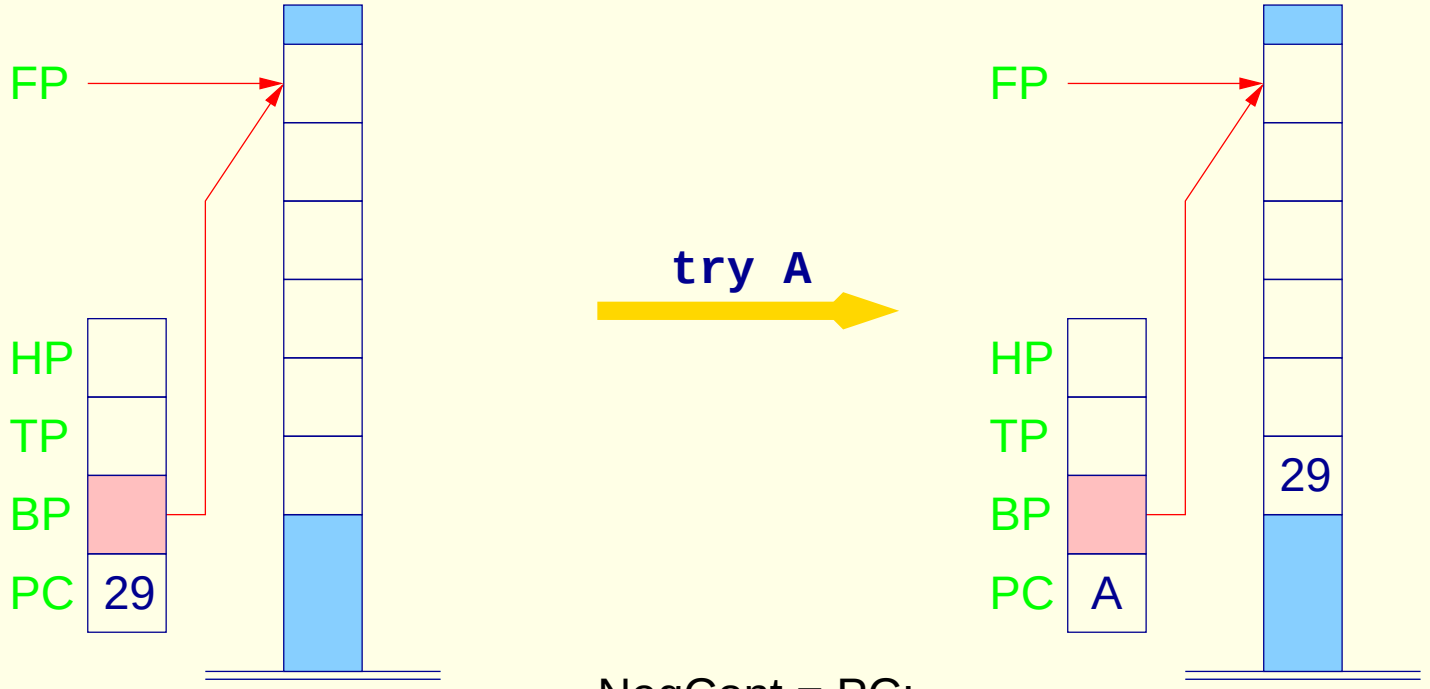
Käsk `setbtp` salvestab registrid HP, TP, BP




```
HPold = HP;
TPold = TP;
BPold = BP;
BP = FP;
```

Predikaadid

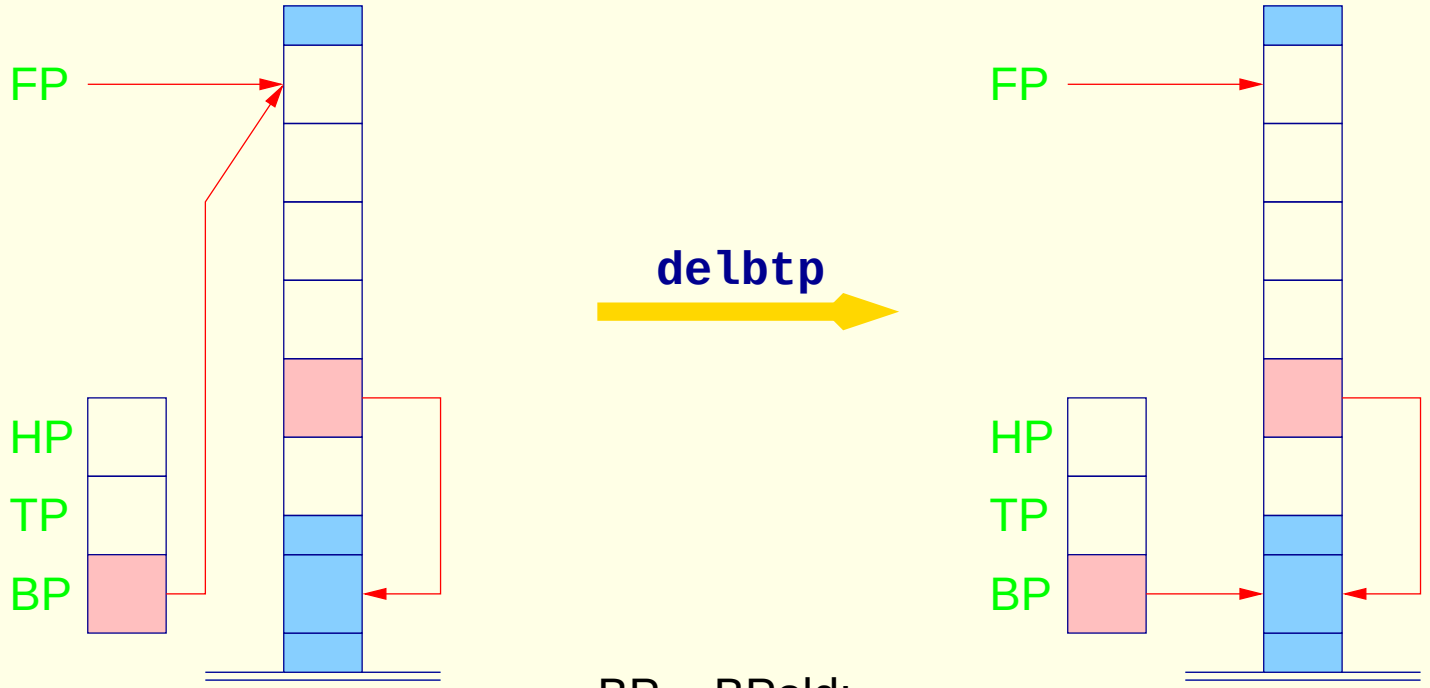
Käsk `try A` ebaõnnestumisel ”proovitava” jätku aadressi ja seejärel hüppab reeglile aadrssil A



NegCont = PC;
PC = A;

Predikaadid

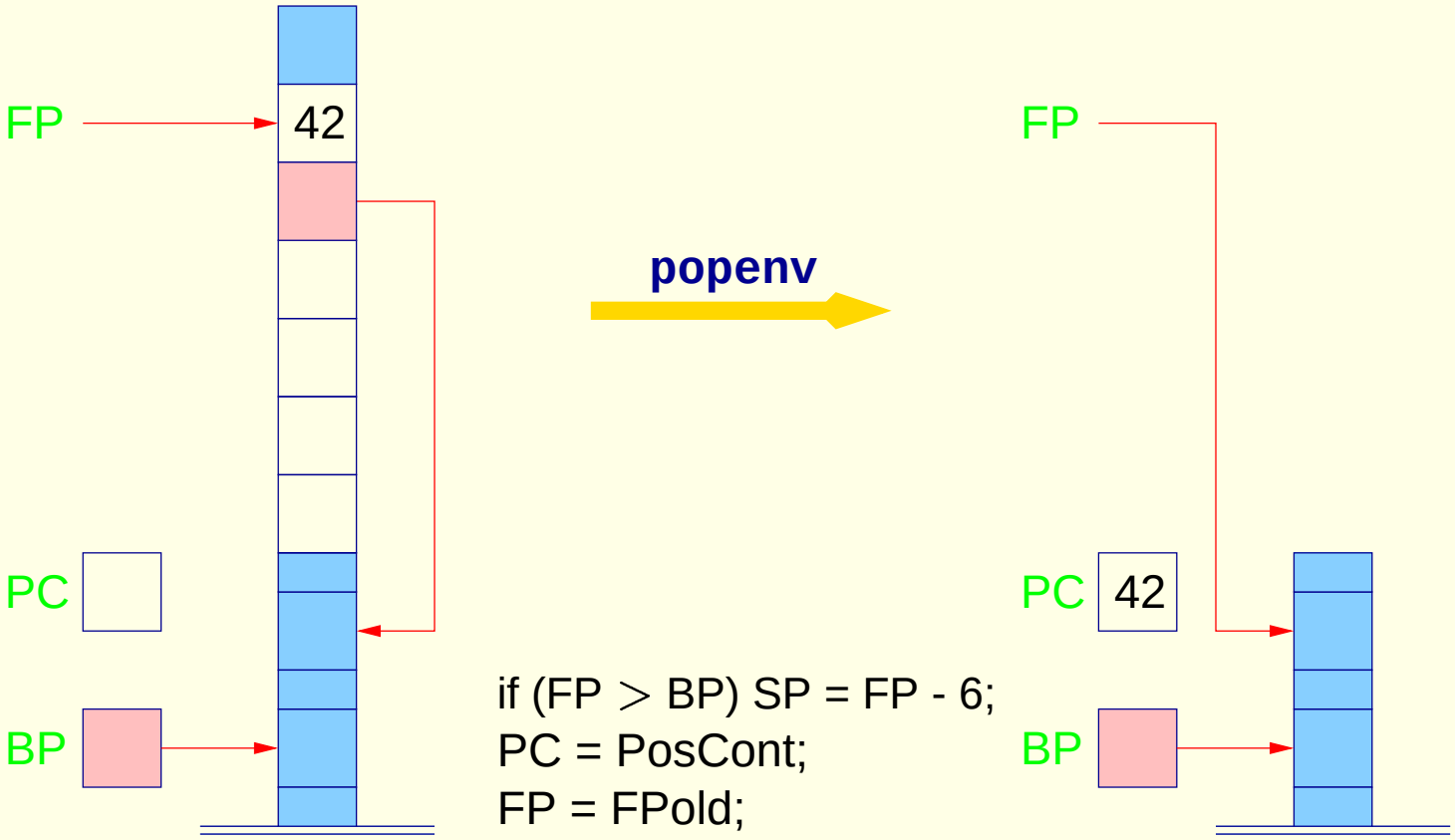
Käsk `delbtp` taastab registri BP väärtuse



BP = BPold;

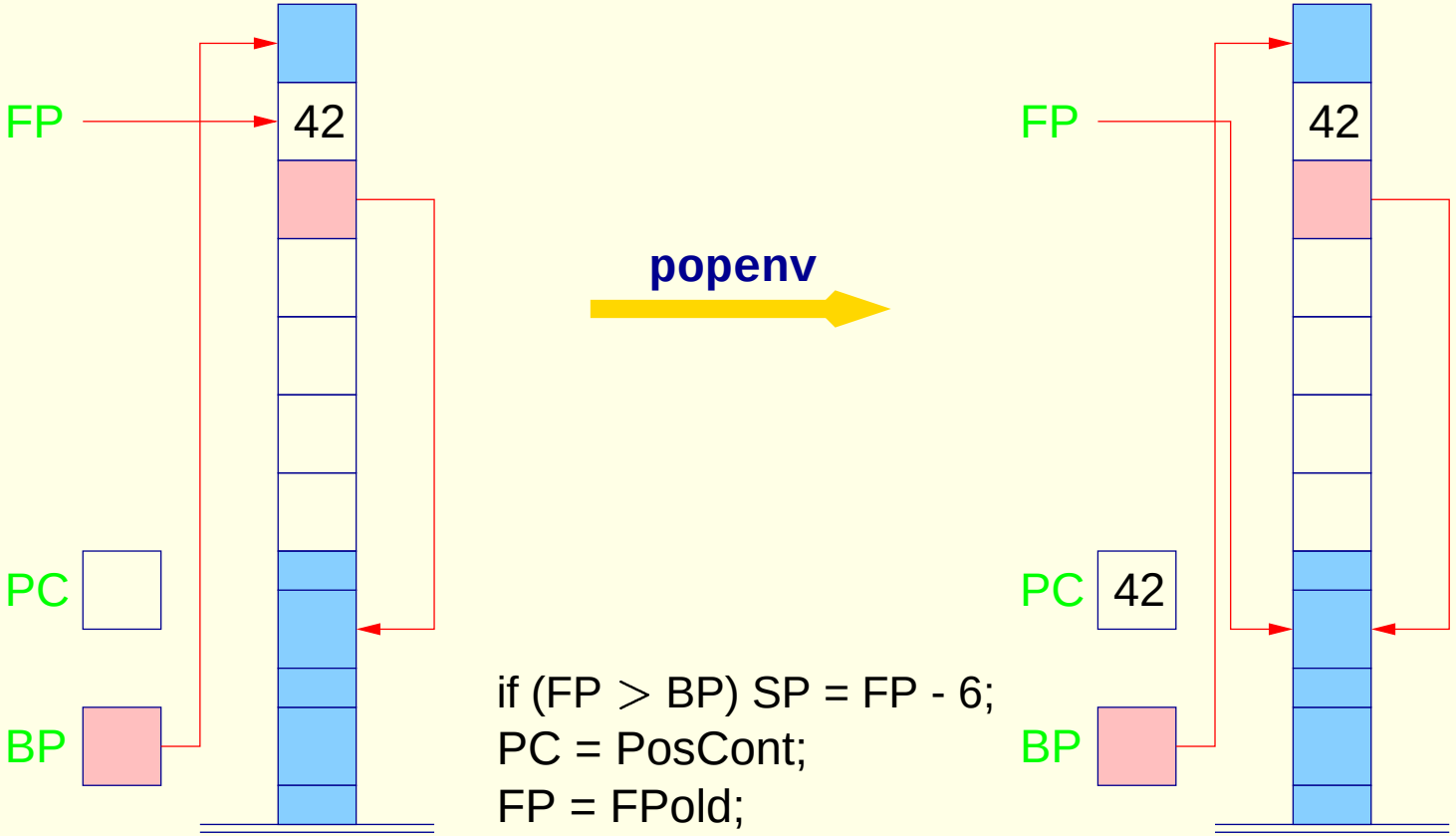
Predikaadid

Käsk popenv taastab registrid FP ja PC ning võimalusel vabastab freimi



Predikaadid

Kui FP $\leq$ BP siis freimi ei vabastata



Päringud ja programmid

- Programmi $p \equiv rr_1 \ldots rr_h ? g$ transleerimisel genereeritakse:
  - päringu g väärtustamisele vastav kood;
  - predikaatide rr_i definitsioonidele vastav kood.
- Päringu väärtustamisele eelneb:
  - registrite initsialiseerimine;
  - globaalsetele muutujatele mälu reserveerimine.
- Päringu väärtustamisele järgneb:
  - globaalsete muutujate väärtuste väljastamine.

Päringud ja programmid

$$
\begin{array}{lcll}
\text{code}\ (rr_1 \ldots rr_h\, ?g) & = & \texttt{init} & \text{code}_P\ rr_1 \\
 & & \texttt{pushenv d} & \ldots \\
 & & \text{code}_G\ g\ \rho & \text{code}_P\ rr_h \\
 & & \texttt{halt d} &
\end{array}
$$

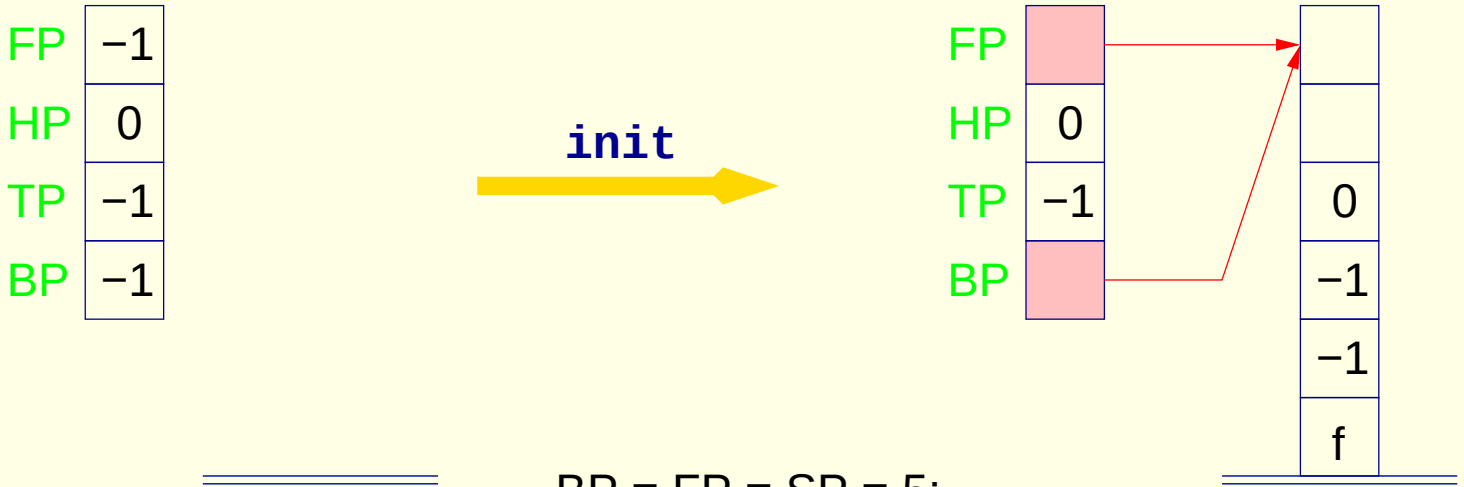
kus $free(g) = \{X_1, \ldots, X_d\}$ ja $\rho = \{X_i \mapsto i \mid i = 1 \ldots d\}$.

Käsk `halt d` ...

- ... lõpetab programmi töö;
- ... väljastab globaalsete muutujate väärtused;
- ... kasutaja nõudel teostab tagasipöördumise.

Päringud ja programmid

Käsk `init` loob algse tagasipöördumispunkti




```
BP = FP = SP = 5;
S[0] = f;
S[1] = S[2] = −1;
S[3] = 0;
BP = FP;
```

Päringu *g* ebaõnnestumisel täidetav kood (aadressil f) võib näiteks välja trükkida teate ebaõnnestumisest.

Päringud ja programmid

Näide: $\mathtt{t}(X) \leftarrow \bar{X} = b$ $\quad \mathtt{q}(X) \leftarrow \mathtt{s}(\bar{X})$ $\quad \mathtt{s}(X) \leftarrow \bar{X} = a$

$\mathtt{p} \leftarrow \mathtt{q}(X), \mathtt{t}(\bar{X})$ $\quad \mathtt{s}(X) \leftarrow \mathtt{t}(\bar{X})$ $\quad ?\,\mathtt{p}$

```
    init          p/0: pushenv 1     q/1: pushenv 1     E: pushenv 1
    pushenv 0          mark B             mark D           mark G
    mark A             putvar 1           putref 1         putref 1
    call p/0           call q/1           call s/1         call t/1
 A: halt 0          B: mark C          D: popenv        G: popenv
t/1: pushenv 1          putref 1        s/1: setbtp      F: pushenv 1
    putref 1           call t/1           try E            putref 1
    uatom b        C: popenv              delbtp           uatom a
    popenv                                jump F           popenv
```