Programmeerimiskeelde massiivide lisamiseks loeme, et muutujate hulk $\mathbf{Var}$ on tükeldatud kaheks — $\mathbf{Var}_{\mathrm{sk}}$ ja $\mathbf{Var}_{\mathrm{m}}$.

- $\mathbf{Var}_{\mathrm{sk}}$ — täisarvutüüpi muutujad.
- $\mathbf{Var}_{\mathrm{m}}$ — muutujad, mille väärtusteks on ühemõõtmelised täisarvumassiivid.

Loeme, et massiivide rajad on $-\infty..\infty$.

- S.t. massiivide piiridest väljastpoolt lugemist / kirjutamist me ei uuri.
- Soovi korral võime rajadesse jäämise kontrolli meie uuritavatesse programmidesse lisada.

Programmiolek s pidi igale muutujale omistama tema väärtuse.

Nüüd koosneb s kahest osast: $s = (s_{\mathrm{sk}}, s_{\mathrm{m}})$.

$$s_{\mathrm{sk}} : \mathbf{Var}_{\mathrm{sk}} \to \mathbb{Z}$$
$$s_{\mathrm{m}} : \mathbf{Var}_{\mathrm{m}} \to \mathbb{Z} \to \mathbb{Z}$$

- s_{sk} annab igale täisarvutüüpi muutujale tema väärtuse.
- s_{m} annab igale massiivitüüpi muutujale a ja indeksile n välja $a[n]$ väärtuse — $s_{\mathrm{m}}(a)(n)$.

Aritmeetilist avaldist A defineeriv abstraktne süntaks on järgmine:

$$
\begin{array}{rcl}
A & ::= & n \\
& | & x \\
& | & -A_1 \\
& | & A_1 + A_2 \mid A_1 - A_2 \mid A_1 * A_2 \mid A_1 / A_2 \\
& | & a[A_1]
\end{array}
$$

kus A_1, A_2 on aritmeetilised avaldised, $x \in \mathbf{Var}_{\mathrm{sk}}$, $a \in \mathbf{Var}_{\mathrm{m}}$ ja $n \in \mathbb{Z}$.

Tõeväärtusavaldist B defineeriv abstraktne süntaks on järgmine:

$$\begin{array}{rcl} B & ::= & \mathsf{true} \mid \mathsf{false} \\ & \mid & A_1 = A_2 \mid A_1 \leqslant A_2 \mid \ldots \\ & \mid & \neg B_1 \\ & \mid & B_1 \wedge B_2 \mid B_1 \vee B_2 \mid \ldots \end{array}$$

kus A_1, A_2 on aritmeetilised avaldised, B_1 ja B_2 tõeväärtusavaldised.

Programmi S defineeriv abstraktne süntaks on järgmine:

$$
\begin{array}{rcl}
S & ::= & x := A \\
 & | & a[A_1] := A_2 \\
 & | & \mathit{skip} \\
 & | & S_1; S_2 \\
 & | & \mathit{if}\ B\ \mathit{then}\ S_1\ \mathit{else}\ S_2 \\
 & | & \mathit{while}\ B\ \mathit{do}\ S_1
\end{array}
$$

kus $x \in \mathbf{Var}_{\mathrm{sk}}$, $a \in \mathbf{Var}_{\mathrm{m}}$, A, A_1, A_2 on aritmeetilised avaldised, B tõeväärtusavaldis ning S_1 ja S_2 programmid.

Aritmeetilisele avaldisele A ja programmiolekule s seame vastavusse täisarvu $\mathcal{A}[\![A]\!]s$.

Funktsiooni $\mathcal{A}$ tüüp on

$$\mathbf{Aexp} \to (\mathbf{State} \to \mathbb{Z}),$$

$$\begin{aligned}
\mathcal{A}[\![n]\!]s &:= n \\
\mathcal{A}[\![x]\!]s &:= s_{\mathrm{sk}}(x) \\
\mathcal{A}[\![-A]\!]s &:= -\mathcal{A}[\![A]\!]s \\
\mathcal{A}[\![A_1 + A_2]\!]s &:= \mathcal{A}[\![A_1]\!]s + \mathcal{A}[\![A_2]\!]s \\
\mathcal{A}[\![A_1 - A_2]\!]s &:= \mathcal{A}[\![A_1]\!]s - \mathcal{A}[\![A_2]\!]s \\
\mathcal{A}[\![a[A_1]]\!]s &:= s_{\mathrm{m}}(a)(\mathcal{A}[\![A_1]\!]s) \\
&\ldots
\end{aligned}$$

Tõeväärtusavaldisele B ja programmiolekule s seame vastavusse tõeväärtuse $\mathcal{B}[\![B]\!]s$.

Funktsiooni $\mathcal{B}$ tüüp on

$$\mathbf{Bexp} \to (\mathbf{State} \to \mathbb{B})$$

$$\begin{aligned}
\mathcal{B}[\![\mathsf{true}]\!]s &:= \mathsf{true} \\
\mathcal{B}[\![A_1 = A_2]\!]s &:= \begin{cases} \mathsf{true}, & \text{kui } \mathcal{A}[\![A_1]\!]s = \mathcal{A}[\![A_2]\!]s \\ \mathsf{false}, & \text{muidu} \end{cases} \\
\mathcal{B}[\![\neg B]\!]s &:= \neg \mathcal{B}[\![B]\!]s \\
\mathcal{B}[\![B_1 \wedge B_2]\!]s &:= \mathcal{B}[\![B_1]\!]s \wedge \mathcal{B}[\![B_2]\!]s \\
&\ldots
\end{aligned}$$

Semantika:

$$\langle x := A, s\rangle \Longrightarrow (s_{\mathrm{sk}}\big[x \mapsto \mathcal{A}[\![A]\!]s\big], s_{\mathrm{m}})$$

Teatavasti $s = (s_{\mathrm{sk}}, s_{\mathrm{m}})$.

Massiivielemendile omistamine:

$$\langle a[A_1] := A_2, s\rangle \Longrightarrow (s_{\mathrm{sk}}, s_{\mathrm{m}}\big[a \mapsto a\big[\mathcal{A}[\![A_1]\!]s \mapsto \mathcal{A}[\![A_2]\!]s\big]\big]) \ .$$

Teiste konstruktsioonide semantika jääb samaks.

Kui a on mingi massiiv ning i ja e aritmeetilised avaldised, siis tähistagu $(a; i : e)$ massiivi, mille väärtus kohal i on e ja mis muidu on võrdne a-ga. Siis

$$\langle a[A_1] := A_2, s\rangle \Longrightarrow (s_{\mathrm{sk}}, s_{\mathrm{m}}\big[a \mapsto (s_{\mathrm{m}}(a); \mathcal{A}[\![A_1]\!]s : \mathcal{A}[\![A_2]\!]s)\big]) \ .$$

$(a; i : e)$ jaoks kehtivad:

- $(a; i : e)[i] = e$.
- $(a; i : e)[j] = a[j]$, kui meil on teada, et $i \neq j$.
- $(a; i : a[i]) = a$.

Aksioom, mis väljendab massiivielemendile omistamise osalist korrektsust:

$$\{Q^a_{(a;i:e)}\}a[i] := e\{Q\}$$

S.t. eeltingimuses tuleb järeltingimusele Q vastavas valemis a asendada $(a; i : e)$-ga.

Termineeruvus:

$$\{\mathsf{true}\}a[i] := e \downarrow$$

Näide: sorteerimine pistemeetodil.

```
j := 2
while j ⩽ n do
    k := a[j]
    i := j
    while i > 1 ∧ a[i − 1] > k do
        a[i] := a[i − 1]
        i := i − 1
    a[i] := k
    j := j + 1
```

S.t. sorteeritakse massiivi a lõik $1..n$.

Toome sisse järgmised tähistused valemites kasutamiseks:

- $sort_m(a)$ tähendagu seda, et massiivi a lõik $1..m$ on sorteeritud.
  - $1 \leqslant l \leqslant m-1 \Rightarrow a[l] \leqslant a[l+1]$.
  - Kui $m_1 \leqslant m_2$, siis $sort_{m_2}(a) \Rightarrow sort_{m_1}(a)$.
- $a \approx_m b$ tähendagu seda, et massiivi a lõik $1..m$ on massiivi b lõigu $1..m$ mingi permutatsioon.

Eel- ja järeltingimus:

$$a_0 = a$$

$j := 2$
$\mathit{while}\ j \leqslant n\ \mathit{do}$
$\quad k := a[j]$
$\quad i := j$
$\quad \mathit{while}\ i > 1 \wedge a[i-1] > k\ \mathit{do}$
$\quad\quad a[i] := a[i-1]$
$\quad\quad i := i - 1$
$\quad a[i] := k$
$\quad j := j + 1$

$$a \approx_n a_0 \wedge \mathit{sort}_n(a)$$

$a_0 = a$

$j := 2$

$a \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2$

$while\ j \leqslant n\ do$

$\quad k := a[j]$

$\quad i := j$

$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge (i < j \Rightarrow a[j-1] \leqslant a[j])$

$\quad while\ i > 1 \wedge a[i-1] > k\ do$

$\quad\quad a[i] := a[i-1]$

$\quad\quad i := i - 1$

$\quad a[i] := k$

$\quad j := j + 1$

$a \approx_n a_0 \wedge sort_n(a)$

Vaatame välimist tsüklit.

Invariant R on $a \approx_n a_0 \wedge sort_{j-1}(a)$.

Vaja näidata, et väitest $a \approx_n a_0 \wedge sort_{j-1}(a) \wedge j > n$ järeldub $a \approx_n a_0 \wedge sort_n(a)$.

See on ilmne.

$$a \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge j \leqslant n$$

$k := a[j]$

$i := j$

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge \big(i < j \Rightarrow a[j-1] \leqslant a[j]\big)$$

$while\ i > 1 \wedge a[i-1] > k\ do$

$\quad a[i] := a[i-1]$

$\quad i := i - 1$

$a[i] := k$

$j := j + 1$

$$a \approx_n a_0 \wedge sort_{j-1}(a)$$

$$a \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant j \leqslant n$$

```
k := a[j]
i := j
```

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge (i < j \Rightarrow a[j-1] \leqslant a[j])$$

```
while i > 1 ∧ a[i − 1] > k do
    a[i] := a[i − 1]
    i := i − 1
a[i] := k
```

$$a \approx_n a_0 \wedge sort_j(a)$$

```
j := j + 1
```

$$a \approx_n a_0 \wedge sort_{j-1}(a)$$

$a \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant j \leqslant n$

$k := a[j]$

$i := j$

$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge \big(i < j \Rightarrow a[j-1] \leqslant a[j]\big)$

$while\ i > 1 \wedge a[i-1] > k\ \ do$

$\quad a[i] := a[i-1]$

$\quad i := i - 1$

$(a; i : k) \approx_n a_0 \wedge sort_j((a; i : k))$

$a[i] := k$

$a \approx_n a_0 \wedge sort_j(a)$

$j := j + 1$

$a \approx_n a_0 \wedge sort_{j-1}(a)$

Vaatame sisemist tsüklit.

Invariant R on

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge \\ \big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \ .$$

Meil tuleb näidata, et väitest

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge \\ \big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge \big(i \leqslant 1 \vee a[i-1] \leqslant k\big)$$

järeldub $(a; i : k) \approx_n a_0$ ja $sort_j((a; i : k))$.

Neist kahest väitest esimene on kohe antud.

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge \big(i \leqslant 1 \vee a[i-1] \leqslant k\big)$$

$sort_j((a; i : k))$ tuletamiseks vaatame kolme võimalust.

- $i = 1$. Siit saame

  1. $i < j$;
  2. $a[j-1] \leqslant a[j]$;
  3. $sort_j(a)$;
  4. $k \leqslant a[1] \leqslant a[2]$;
  5. $sort_j((a; 1 : k))$.

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge \\ \big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge \big(i \leqslant 1 \vee a[i-1] \leqslant k\big)$$

- $1 < i < j$. Siit saame
  1. $a[j-1] \leqslant a[j]$;
  2. $sort_j(a)$;
  3. $a[i-1] \leqslant k \leqslant a[i] \leqslant a[i+1]$;
  4. $sort_j((a; i : k))$.

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge \big(i \leqslant 1 \vee a[i-1] \leqslant k\big)$$

- $1 < i = j$. Siit saame
  1. $a[j-1] \leqslant k$.
  2. $sort_j((a; j : k))$.

Vaatame sisemist tsüklit.

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge \\ \big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge i > 1 \wedge a[i-1] > k$$

6 $\quad a[i] := a[i-1]$

7 $\quad i := i - 1$

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge \\ \big(i < j \Rightarrow a[j-1] \leqslant a[j]\big)$$

$$(a;i:k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge i > 1 \wedge a[i-1] > k$$

$$((a;i:a[i-1]);i-1:k) \approx_n a_0 \wedge sort_{j-1}((a;i:a[i-1])) \wedge$$
$$j \geqslant 2 \wedge 1 \leqslant i-1 \leqslant j \leqslant n \wedge (a;i:a[i-1])[i-1] \geqslant k \wedge$$
$$\big(i-1 < j \Rightarrow (a;i:a[i-1])[j-1] \leqslant (a;i:a[i-1])[j]\big)$$

6 $\quad a[i] := a[i-1]$

$$(a;i-1:k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i-1 \leqslant j \leqslant n \wedge$$
$$a[i-1] \geqslant k \wedge \big(i-1 < j \Rightarrow a[j-1] \leqslant a[j]\big)$$

7 $\quad i := i-1$

$$(a;i:k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big)$$

Punasest peab järelduma sinine...

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge i > 1 \wedge a[i-1] > k$$

$$((a; i : a[i-1]); i-1 : k) \approx_n a_0 \wedge sort_{j-1}((a; i : a[i-1])) \wedge$$
$$j \geqslant 2 \wedge 1 \leqslant i-1 \leqslant j \leqslant n \wedge (a; i : a[i-1])[i-1] \geqslant k \wedge$$
$$\big(i-1 < j \Rightarrow (a; i : a[i-1])[j-1] \leqslant (a; i : a[i-1])[j]\big)$$

Lihtsustame...

$$(a;i:k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$

$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge i > 1 \wedge a[i-1] > k$$

$$((a;i:a[i-1]);i-1:k) \approx_n a_0 \wedge sort_{j-1}((a;i:a[i-1])) \wedge$$

$$j \geqslant 2 \wedge 1 \leqslant i-1 \leqslant j \leqslant n \wedge (a;i:a[i-1])[i-1] \geqslant k \wedge$$

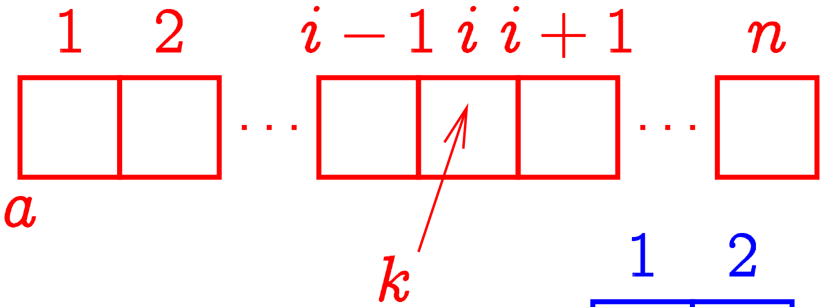
$$\big(i-1 < j \Rightarrow (a;i:a[i-1])[j-1] \leqslant (a;i:a[i-1])[j]\big)$$

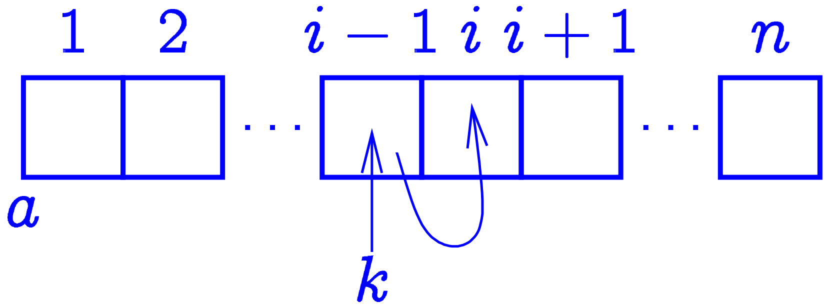
$$(a;i:k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge a[i-1] > k$$
$$((a;i:a[i-1]);i-1:k) \approx_n a_0 \wedge sort_{j-1}((a;i:a[i-1])) \wedge$$
$$j \geqslant 2 \wedge 1 \leqslant i-1 \leqslant j \leqslant n \wedge a[i-1] \geqslant k \wedge$$
$$\big(i-1 < j \Rightarrow (a;i:a[i-1])[j-1] \leqslant (a;i:a[i-1])[j]\big)$$

Vaatame kõik sinised väited läbi. . .

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge a[i-1] > k$$

$$((a; i : a[i-1]); i-1 : k) \approx_n a_0$$

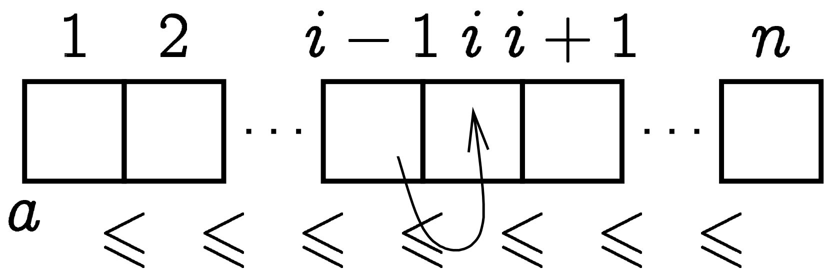




$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$

$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge a[i-1] > k$$

$$sort_{j-1}((a; i : a[i-1]))$$



$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge \boxed{2 \leqslant i \leqslant j \leqslant n} \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge a[i-1] > k$$
$$j \geqslant 2 \wedge 1 \leqslant i - 1 \leqslant j \leqslant n$$

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$

$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge \boxed{a[i-1] > k}$$

$$a[i-1] \geqslant k$$

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge \boxed{2 \leqslant i \leqslant j \leqslant n} \wedge a[i] \geqslant k \wedge$$

$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge a[i-1] > k$$

$$\boxed{i - 1 < j} \Rightarrow (a; i : a[i-1])[j-1] \leqslant (a; i : a[i-1])[j]$$

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\bigl(i < j \Rightarrow a[j-1] \leqslant a[j]\bigr) \wedge a[i-1] > k$$
$$(a; i : a[i-1])[j-1] \leqslant (a; i : a[i-1])[j]$$

Vaatame kolme võimalust

1. Kui $i < j - 1$, siis
   1. $a[j-1] \leqslant a[j]$;
   2. $(a; i : a[i-1])[j-1] = a[j-1]$;
   3. $(a; i : a[i-1])[j] = a[j]$.

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge a[i-1] > k$$
$$(a; i : a[i-1])[j-1] \leqslant (a; i : a[i-1])[j]$$

Vaatame kolme võimalust

2. Kui $i = j - 1$, siis
   1. $a[i-1] = a[j-2] \leqslant a[j-1] \leqslant a[j]$;
   2. $(a; i : a[i-1])[j-1] = a[i-1]$;
   3. $(a; i : a[i-1])[j] = a[j]$.

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge$$
$$\big(i < j \Rightarrow a[j-1] \leqslant a[j]\big) \wedge a[i-1] > k$$
$$(a; i : a[i-1])[j-1] \leqslant (a; i : a[i-1])[j]$$

Vaatame kolme võimalust

3. Kui $i = j$, siis

   1. $(a; i : a[i-1])[j-1] = a[j-1] = a[i-1]$;
   2. $(a; i : a[i-1])[j] = a[i-1]$.

$a \approx_n a_0 \land sort_{j-1}(a) \land 2 \leqslant j \leqslant n$

$k := a[j]$

$i := j$

$(a; i : k) \approx_n a_0 \land sort_{j-1}(a) \land j \geqslant 2 \land 1 \leqslant i \leqslant j \leqslant n \land a[i] \geqslant k \land \big(i < j \Rightarrow a[j-1] \leqslant a[j]\big)$

$while\ i > 1 \land a[i-1] > k\ do$

  $a[i] := a[i-1]$

  $i := i - 1$

$(a; i : k) \approx_n a_0 \land\ sort_j((a; i : k))$

$a[i] := k$

$a \approx_n a_0 \land sort_j(a)$

$j := j + 1$

$a \approx_n a_0 \land sort_{j-1}(a)$

$$a \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant j \leqslant n$$

$$k := a[j]$$

$$(a; j : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant j \leqslant j \leqslant n \wedge a[j] \geqslant k \wedge \left(j < j \Rightarrow a[j-1] \leqslant a[j]\right)$$

$$i := j$$

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge \left(i < j \Rightarrow a[j-1] \leqslant a[j]\right)$$

$$while\ i > 1 \wedge a[i-1] > k\ \ do$$

$$\quad a[i] := a[i-1]$$

$$\quad i := i - 1$$

$$(a; i : k) \approx_n a_0 \wedge\ sort_j((a; i : k))$$

$$a[i] := k$$

$$a \approx_n a_0 \wedge sort_j(a)$$

$$j := j + 1$$

$$a \approx_n a_0 \wedge sort_{j-1}(a)$$

$a \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant j \leqslant n$

$k := a[j]$

$(a; j : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant j \leqslant n \wedge a[j] \geqslant k$

$i := j$

$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge \big(i < j \Rightarrow a[j-1] \leqslant a[j]\big)$

$while\ i > 1 \wedge a[i-1] > k\ do$

$\quad a[i] := a[i-1]$

$\quad i := i - 1$

$(a; i : k) \approx_n a_0 \wedge\ sort_j((a; i : k))$

$a[i] := k$

$a \approx_n a_0 \wedge sort_j(a)$

$j := j + 1$

$a \approx_n a_0 \wedge sort_{j-1}(a)$

$$a \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant j \leqslant n$$

$$(a; j : a[j]) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant j \leqslant n \wedge a[j] \geqslant a[j]$$

$k := a[j]$

$$(a; j : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge 2 \leqslant j \leqslant n \wedge a[j] \geqslant k$$

$i := j$

$$(a; i : k) \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2 \wedge 1 \leqslant i \leqslant j \leqslant n \wedge a[i] \geqslant k \wedge (i < j \Rightarrow a[j-1] \leqslant a[j])$$

$\mathit{while}\ i > 1 \wedge a[i-1] > k\ \mathit{do}$

$\quad a[i] := a[i-1]$

$\quad i := i - 1$

$$(a; i : k) \approx_n a_0 \wedge sort_j((a; i : k))$$

$a[i] := k$

$$a \approx_n a_0 \wedge sort_j(a)$$

$j := j + 1$

$$a \approx_n a_0 \wedge sort_{j-1}(a)$$

On ilmne, et punasest järeldub sinine.

$$a_0 = a$$

$$\begin{array}{l}
j := 2 \\
\mathit{while}\ j \leqslant n\ \mathit{do} \\
\quad k := a[j] \\
\quad i := j \\
\quad \mathit{while}\ i > 1 \wedge a[i-1] > k\ \mathit{do} \\
\quad\quad a[i] := a[i-1] \\
\quad\quad i := i - 1 \\
\quad a[i] := k \\
\quad j := j + 1
\end{array}$$

$$a \approx_n a_0 \wedge \mathit{sort}_{j-1}(a) \wedge j \geqslant 2$$

$$a \approx_n a_0 \wedge \mathit{sort}_n(a)$$

$a_0 = a$

$a \approx_n a_0 \wedge sort_1(a) \wedge 2 \geqslant 2$

$a \approx_n a_0 \wedge sort_{j-1}(a) \wedge j \geqslant 2$

```
j := 2
while j ⩽ n do
    k := a[j]
    i := j
    while i > 1 ∧ a[i − 1] > k do
        a[i] := a[i − 1]
        i := i − 1
    a[i] := k
    j := j + 1
```

$a \approx_n a_0 \wedge sort_n(a)$

On ilmne, et punasest järeldub sinine.

Lisame keelde protseduurid. . .

Eristame kolme erinevat sorti muutujaid — lokaalsed muutujad, globaalsed muutujad ja protseduuriparameetrid.

- Loeme, et kõik protseduurid kasutavad oma lokaalsete muutujate ja parameetrite jaoks samu nimesid.
  - võtame lihtsalt kasutatavate muutujanimede ühendi. . .
- Parameetri ja lokaalse muutuja erinevus — parameetrile ei saa protseduuri kehas omistada.

Koos eelneva jaotusega skalaarideks ja massiivideks annab see meile kuus erinevat klassi — $\mathbf{Var}_{\mathrm{lsk}}$, $\mathbf{Var}_{\mathrm{gsk}}$, $\mathbf{Var}_{\mathrm{psk}}$, $\mathbf{Var}_{\mathrm{lm}}$, $\mathbf{Var}_{\mathrm{gm}}$ ja $\mathbf{Var}_{\mathrm{pm}}$.

Olgu $\mathbf{PName}$ protseduurinimede hulk.

Funktsioon $\mathbf{K} : \mathbf{PName} \rightarrow \mathbf{Prog}$ seadku igale protseduurinimele vastavusse selle protseduuri keha.

Protseduuri M väljakutseks kasutame süntaksit

$$call\ M(p_1 := A_1, p_2 := A_2, \ldots, p_k := A_k)$$

kus $p_1, \ldots, p_k$ on protseduuriparameetrid ja $A_1, \ldots, A_k$ aritmeetilised avaldised või massiivitüüpi muutujad.

- M-i väljakutsel saab p_i väärtuseks A_i väärtus.
- Ülejäänud parameetrid ja kõik lokaalsed muutujad saavad mingi vaikeväärtuse.

Programmi *konfiguratsioon* koosneb

- parasjagu pooleliolevate protseduuride konfiguratsioonide järjendist (pinust);
- globaalsete muutujate väärtustest.

Programm lõpetab töö, kui protseduurikonfiguratsioonide pinu tühjaks saab. Siis jäävad järgi ainult globaalsete muutujate väärtused.

„Globaalsete muutujate väärtused“ on funktsioonide paar $s_{\mathrm{g}} = (s_{\mathrm{gsk}}, s_{\mathrm{gm}})$, kus

$$
\begin{aligned}
s_{\mathrm{gsk}} &: \mathbf{Var}_{\mathrm{gsk}} \to \mathbb{Z} \\
s_{\mathrm{gm}} &: \mathbf{Var}_{\mathrm{gm}} \to \mathbb{Z} \to \mathbb{Z} \quad .
\end{aligned}
$$

„Lokaalsete muutujate väärtused“ on funktsioonide paar $s_{\mathrm{l}} = (s_{\mathrm{lsk}}, s_{\mathrm{lm}}, s_{\mathrm{psk}}, s_{\mathrm{pm}})$, kus

$$\begin{aligned} s_{\mathrm{lsk}} &: \mathbf{Var}_{\mathrm{lsk}} \to \mathbb{Z} \\ s_{\mathrm{lm}} &: \mathbf{Var}_{\mathrm{lm}} \to \mathbb{Z} \to \mathbb{Z} \\ s_{\mathrm{psk}} &: \mathbf{Var}_{\mathrm{psk}} \to \mathbb{Z} \\ s_{\mathrm{pm}} &: \mathbf{Var}_{\mathrm{pm}} \to \mathbb{Z} \to \mathbb{Z} \quad . \end{aligned}$$

Protseduuri olek on paar $\langle S, s_{\mathrm{l}} \rangle$, kus S on programm.

Või lihtsalt s_{l}, kui protseduur on oma tööălõpetanud.

Programmid $S \in \mathbf{Prog}$ on

$$\begin{array}{rcl} S & ::= & x := A \\ & | & a[A'] := A'' \\ & | & \mathit{skip} \\ & | & S_1; S_2 \\ & | & \mathit{if}\ B\ \mathit{then}\ S_1\ \mathit{else}\ S_2 \\ & | & \mathit{while}\ B\ \mathit{do}\ S_1 \\ & | & \mathit{call}\ M(p_1 := A_1, p_2 := A_2, \ldots, p_k := A_k) \end{array}$$

kus $x \in \mathbf{Var}_{\mathrm{lsk}} \cup \mathbf{Var}_{\mathrm{gsk}}$, $p_1, \ldots, p_k \in \mathbf{Var}_{\mathrm{psk}} \cup \mathbf{Var}_{\mathrm{pm}}$, $a \in \mathbf{Var}_{\mathrm{lm}} \cup \mathbf{Var}_{\mathrm{gm}}$, A, A', A'' on aritmeetilised avaldised, $A_1, \ldots, A_k$ kas aritmeetilised avaldised või muutujad hulgast $\mathbf{Var}_{\mathrm{lm}} \cup \mathbf{Var}_{\mathrm{gm}}$, B tõeväärtusavaldis ning S_1 ja S_2 programmid.

Ühe sammu tegemine on defineeritud järgmiselt. Olgu $\mathbf{P}$ mingi protseduuriolekute järjend, s.t. $\mathbf{P} = \langle S_1, s_1\rangle\langle S_2, s_2\rangle \cdots \langle S_k, s_k\rangle$, kus $s_1, \ldots, s_k$ on lokaalsete muutujate väärtused.

Mõni S_i võib ka puududa...

$$(\langle x := A, s\rangle\mathbf{P}, s_{\mathrm{g}}) \Longrightarrow (\langle s[x \mapsto \mathcal{A}[\![A]\!](s, s_{\mathrm{g}})]\rangle\mathbf{P}, s_{\mathrm{g}}),$$

kui $x \in \mathbf{Var}_{\mathrm{lsk}}$.

$$(\langle x := A, s\rangle\mathbf{P}, s_{\mathrm{g}}) \Longrightarrow (\langle s\rangle\mathbf{P}, s_{\mathrm{g}}[x \mapsto \mathcal{A}[\![A]\!](s, s_{\mathrm{g}})]),$$

kui $x \in \mathbf{Var}_{\mathrm{gsk}}$.

$$(\langle s\rangle\mathbf{P}, s_{\mathrm{g}}) \Longrightarrow (\mathbf{P}, s_{\mathrm{g}})$$

$$(\langle \mathit{call}\ M(p_1 := A_1, \ldots, p_k := A_k), s\rangle \mathbf{P}, s_{\mathrm{g}}) \Longrightarrow$$
$$(\langle \mathbf{K}(M), s_0[p_i \mapsto \mathcal{A}[\![A_i]\!](s, s_{\mathrm{g}})]\rangle \langle s\rangle \mathbf{P}, s_{\mathrm{g}}),$$

kus s_0 omistab kõigile lokaalsetele muutujatele ja protseduuriparameetritele vaikeväärtused.

$$\frac{(\langle S_1, s\rangle \mathbf{P}, s_{\mathrm{g}}) \Longrightarrow (\mathbf{P}'\langle s'\rangle \mathbf{P}, s'_{\mathrm{g}})}{(\langle S_1; S_2, s\rangle \mathbf{P}, s_{\mathrm{g}}) \Longrightarrow (\mathbf{P}'\langle S_2, s'\rangle \mathbf{P}, s'_{\mathrm{g}})}$$

$$\frac{(\langle S_1, s\rangle \mathbf{P}, s_{\mathrm{g}}) \Longrightarrow (\mathbf{P}'\langle S'_1, s'\rangle \mathbf{P}, s'_{\mathrm{g}})}{(\langle S_1; S_2, s\rangle \mathbf{P}, s_{\mathrm{g}}) \Longrightarrow (\mathbf{P}'\langle S'_1; S_2, s'\rangle \mathbf{P}, s'_{\mathrm{g}})}$$

Teised konstruktsioonid on sarnased.

return-lauset ei ole. Protseduuridest väärtusi tagastada tuleb globaalsete muutujate kaudu.

Programmi korrektsuse üle arutades nõuame, et *call*-lauses tegelikud parameetrid A_i ei sisaldaks globaalseid muutujaid.

- Vajadusel võib nad eelnevalt lokaalsetesse muutujatesse kopeerida.

$\{P\}S\{Q\}$ tähendab, et

- iga lokaalsete muutujate väärtustuse s jaoks,
- iga globaalsete muutujate väärtustuse s_{g} jaoks,

nii et $P(s, s_{\mathrm{g}})$ on tõene,

- kui $(\langle S, s\rangle, s_{\mathrm{g}}) \stackrel{*}{\Longrightarrow} (\langle s'\rangle, s'_{\mathrm{g}})$,

siis $Q(s', s'_{\mathrm{g}})$ on tõene.

Kui programmikonfiguratsioonis on mitu protseduuri, siis P ja Q ei saa viidata pinus allpool olevate protseduuride lokaalsetele muutujatele.

Protseduuri väljakutse korrektsus:

$$\frac{\{P\}\mathbf{K}(M)\{Q\}}{\{P^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)}\}\mathit{call}\ M(p_1 := A_1,\ldots,p_k := A_k)\{Q^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)}\}}$$

$$\frac{\{P\}\mathbf{K}(M)\downarrow}{\{P^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)}\}\mathit{call}\ M(p_1 := A_1,\ldots,p_k := A_k)\downarrow}$$

Reeglite järeldustes tuleb P-s ja Q-s asendada kõik parameetrid ja lokaalsed muutujad.

- $p_1,\ldots,p_k$ asendada $A_1,\ldots,A_k$-ga;
- ülejäänud asendada vaikeväärtustega.

Toodud reeglites toimuv vastab protseduuri M *inline*'imisele.

Need reeglid räägivad ainult $\mathbf{K}(M)$ eel- ja järeltingimuste teisendamisest.

Peale selle kehtib

$$\frac{\{P\}call\ M(p_1 := A_1, \ldots, p_k := A_k)\{Q\}}{\{P \wedge L\}call\ M(p_1 := A_1, \ldots, p_k := A_k)\{Q \wedge L\}}$$

kus L on predikaat, mis ei sisalda globaalseid muutujaid.

Rekursiivsete protseduuride korrektsuse näitamisel pole üle-eelmisel slaidil toodud reeglitest kasu.

Saaksime lõpmata palju alamülesandeid kujul:

> $call\ M(\ldots)$ korrektsuse näitamiseks näita $call\ M(\ldots)$ korrektsust.

Rekursiivsus tähendab tsüklit. Tarvis on mingit tsükliinvarianti.

Protseduuri väljakutse osaline korrektsus:

$$\frac{\{P^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)}\}call\ M(p_1 := A_1,\ldots,p_k := A_k)\{Q^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)}\} \vdash \{P\}\mathbf{K}(M)\{Q\}}{\{P^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)}\}call\ M(p_1 := A_1,\ldots,p_k := A_k)\{Q^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)}\}}$$

Siin $\vdash$ tähendab „on tuletatav“.

S.t. antud kohas $\{P\}\mathbf{K}(M)\{Q\}$ tuletamisel tohime kasutada täiendavat aksioomi.

P ja Q roll on sarnane tsükliinvariandi omaga.

Kui M pole rekursiivne, siis on see reegel samaväärne eelmisega — meil pole seda täiendavat aksioomi kusagil kasutada.

Termineeruvus:

$$\frac{\{P^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)} \wedge (0 \leqslant E^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)} < e)\} \; \mathit{call}\ M(p_1 := A_1, \ldots, p_k := A_k) \downarrow \; \vdash \{P \wedge 0 \leqslant E = e\}\mathbf{K}(M) \downarrow}{\{P^{(p_1,\ldots,p_k,\ldots)}_{(A_1,\ldots,A_k,0\ldots)}\}\mathit{call}\ M(p_1 := A_1, \ldots, p_k := A_k) \downarrow}$$

Siin E on samas rollis mis tsükli termineeruvuse tõestamiselgi.

Reeglite tõestused — induktsiooniga üle üksteise sees olevate lausete $\mathit{call}\ M(p_1 := A_1, \ldots, p_k := A_k)$ arvu.

Näide: vaatame protseduuri ühildussort(a, n), kus $a \in \mathbf{Var}_{\mathrm{pm}}$ ja $n \in \mathbf{Var}_{\mathrm{psk}}$.

Näitame, et iga $X \in \mathbf{Var}_{\mathrm{lm}}$ ja $Y \in \mathbf{Var}_{\mathrm{lsk}}$ jaoks kehtib

$$Y \geqslant 1$$

$$\mathit{call}\ \text{ühildussort}(a := X, n := Y)$$

$$X \approx_Y r \wedge \mathit{sort}_Y(r)$$

kus $r \in \mathbf{Var}_{\mathrm{gm}}$ on mingi fikseeritud globaalne muutuja.

ühildussort(a, n) on

$\mathit{if}\ n = 1\ \mathit{then}$

  $r[1] := a[1]$

else

  $k := \lfloor n/2 \rfloor;\ l := n - k$

  $b := a[1..k];\ c := a[k+1..n]$

  call ühildussort$(a := b, n := k);\ b := r$

  call ühildussort$(a := c, n := l);\ c := r$

  call ühilda$(a := b, b := c, m := k, n := l)$

Eeldame, et kehtib

$$sort_m(a) \wedge sort_n(b) \wedge m \geqslant 1 \wedge n \geqslant 1$$

$$\mathbf{K}(\text{ühilda})$$

$$sort_{m+n}(r) \wedge a[1..m] \parallel b[1..n] \approx_{m+n} r$$

Tarvis näidata:

$n \geqslant 1$

$\mathit{if}\ n = 1\ \mathit{then}$

$\quad r[1] := a[1]$

else

$\quad k := \lfloor n/2 \rfloor;\ l := n - k$

$\quad b := a[1..k];\ c := a[k+1..n]$

$\quad \mathit{call}$ ühildussort$(a := b, n := k);\ b := r$

$\quad \mathit{call}$ ühildussort$(a := c, n := l);\ c := r$

$\quad \mathit{call}$ ühilda$(a := b, b := c, m := k, n := l)$

$a \approx_n r \wedge \mathit{sort}_n(r)$

if-lause *then*-osa:

$n = 1$

2 $r[1] := a[1]$

$a \approx_n r \wedge sort_n(r)$

if-lause *then*-osa:

$$n = 1$$

$$a \approx_n (r; 1 : a[1]) \wedge sort_n((r; 1 : a[1]))$$

2 $r[1] := a[1]$

$$a \approx_n r \wedge sort_n(r)$$

Punasest järeldub sinine.

if-lause *else*-osa:

$n > 1$

$k := \lfloor n/2 \rfloor;\ l := n - k$

$b := a[1..k];\ c := a[k+1..n]$

call ühildussort$(a := b, n := k);\ b := r$

call ühildussort$(a := c, n := l);\ c := r$

call ühilda$(a := b, b := c, m := k, n := l)$

$a \approx_n r \wedge sort_n(r)$

Protseduuri ühilda keha kohta käiv väide lubab meil tuletada

$$sort_k(b) \wedge sort_l(c) \wedge k \geqslant 1 \wedge l \geqslant 1$$

$$\mathit{call}\ \text{ühilda}(a := b, b := c, m := k, n := l)$$

$$sort_{k+l}(r) \wedge b[1..k] \,\|\, c[1..l] \approx_{k+l} r$$

Muudame programmilõigu 4–8 järeltingimust tugevamaks.

$n > 1$

$k := \lfloor n/2 \rfloor;\ l := n - k$

$b := a[1..k];\ c := a[k+1..n]$

call ühildussort$(a := b, n := k);\ b := r$

call ühildussort$(a := c, n := l);\ c := r$

call ühilda$(a := b, b := c, m := k, n := l)$

$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge$

$b[1..k] \,||\, c[1..l] \approx_{k+l} r \wedge \mathit{sort}_{k+l}(r)$

$n > 1$

$k := \lfloor n/2 \rfloor;\ l := n - k$

$b := a[1..k];\ c := a[k+1..n]$

call ühildussort$(a := b, n := k);\ b := r$

call ühildussort$(a := c, n := l);\ c := r$

$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge$

$\mathit{sort}_k(b) \wedge \mathit{sort}_l(c) \wedge k \geqslant 1 \wedge l \geqslant 1$

call ühilda$(a := b, b := c, m := k, n := l)$

$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge$

$b[1..k] \,||\, c[1..l] \approx_{k+l} r \wedge \mathit{sort}_{k+l}(r)$

Vaatame 7. rida...

7 $\quad$ *call* ühildussort($a := c, n := l$)

$$n = k + l \wedge b \approx_k a \wedge r \approx_l a[k+1..n] \wedge$$
$$sort_k(b) \wedge sort_l(r) \wedge k \geqslant 1 \wedge l \geqslant 1$$

$$c := r$$

$$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge$$
$$sort_k(b) \wedge sort_l(c) \wedge k \geqslant 1 \wedge l \geqslant 1$$

Tugevdame ühildussort'i väljakutse järeltingimust...

7 $\quad \mathit{call}$ ühildussort$(a := c, n := l)$

$$n = k + l \wedge b \approx_k a \wedge r \approx_l c \wedge c \approx_l a[k+1..n] \wedge$$
$$sort_k(b) \wedge sort_l(r) \wedge k \geqslant 1 \wedge l \geqslant 1$$

$c := r$

$$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge$$
$$sort_k(b) \wedge sort_l(c) \wedge k \geqslant 1 \wedge l \geqslant 1$$

Rakendame ühildussort'i väljakutse kohta käivat eeldust. . .

$$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge$$
$$sort_k(b) \wedge k \geqslant 1 \wedge l \geqslant 1$$

7 $\quad$ *call* ühildussort$(a := c, n := l)$

$$n = k + l \wedge b \approx_k a \wedge r \approx_l c \wedge c \approx_l a[k+1..n] \wedge$$
$$sort_k(b) \wedge sort_l(r) \wedge k \geqslant 1 \wedge l \geqslant 1$$

$c := r$

$$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge$$
$$sort_k(b) \wedge sort_l(c) \wedge k \geqslant 1 \wedge l \geqslant 1$$

Sama 6. rea jaoks...

$$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge k \geqslant 1 \wedge l \geqslant 1$$

6 $\quad$ *call* ühildussort$(a := b, n := k)$

$$n = k + l \wedge r \approx_k b \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge$$
$$sort_k(r) \wedge k \geqslant 1 \wedge l \geqslant 1$$

$$r \approx_k b \wedge b \approx_k a \Rightarrow r \approx_k a$$

$b := r$

$$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge$$
$$sort_k(b) \wedge k \geqslant 1 \wedge l \geqslant 1$$

$$n > 1$$

$$n = \lfloor n/2 \rfloor + n - \lfloor n/2 \rfloor \wedge a[1..\lfloor n/2 \rfloor] \approx_{\lfloor n/2 \rfloor} a \wedge$$
$$a[\lfloor n/2 \rfloor + 1..n] \approx_{n-\lfloor n/2 \rfloor} a[\lfloor n/2 \rfloor + 1..n] \wedge$$
$$\lfloor n/2 \rfloor \geqslant 1 \wedge n - \lfloor n/2 \rfloor \geqslant 1$$

4 $\quad k := \lfloor n/2 \rfloor;\ l := n - k$

$$n = k + l \wedge a[1..k] \approx_k a \wedge$$
$$a[k+1..n] \approx_l a[k+1..n] \wedge k \geqslant 1 \wedge l \geqslant 1$$

5 $\quad b := a[1..k];\ c := a[k+1..n]$

$$n = k + l \wedge b \approx_k a \wedge c \approx_l a[k+1..n] \wedge k \geqslant 1 \wedge l \geqslant 1$$

Punasest järeldub sinine.

Ühildussorteerimise termineeruvuse näitamiseks — avaldiseks E sobib parameetri n väärtus.