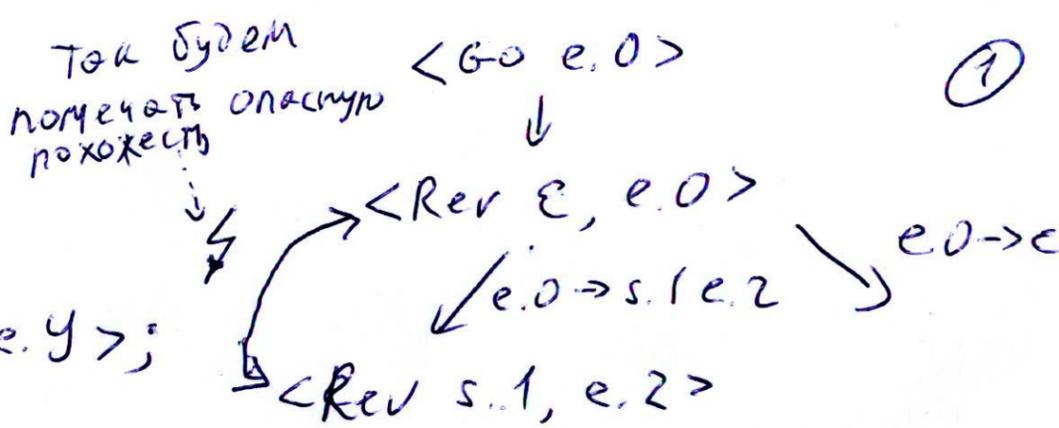
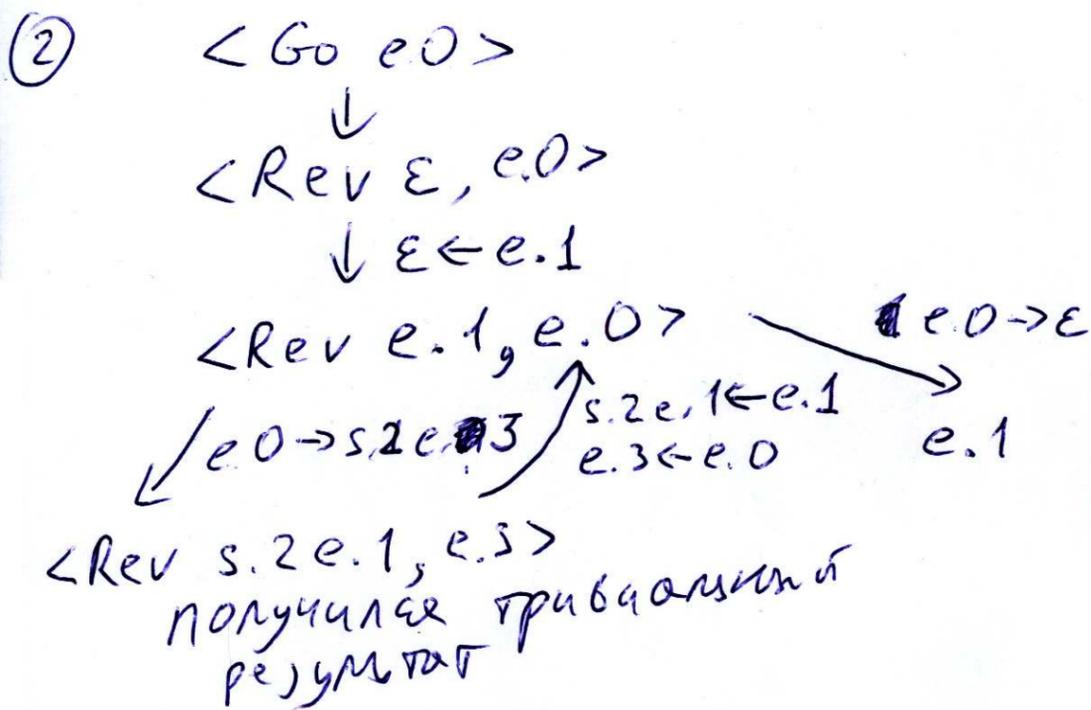


\$ENTRY GO E
 e.x = <Rev E, e.x>

Rev E
 e.A, s.x e.y = <Rev s.x e.A, e.y>;
 e.A, E = e.A;

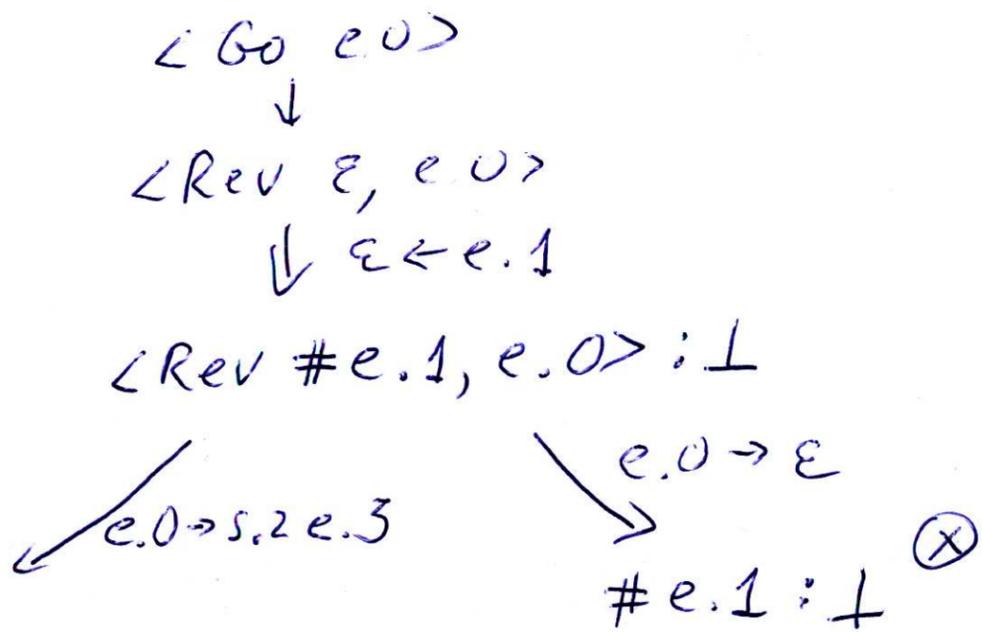


③ А теперь - фокус!



Попробуем заморозить
 аккумулятор #e.1

③ - фокус



<Rev s.2 #e.1, e.3> : ⊥
 (пока (слева!) не будем
 ничего тут рассматривать,
 проигнорируем эту конфигурацию)

здесь формат
 нарушился,
 обобщаем:

#e.1 ⊥ ⊥ = #e.1

Параметры времени компиляции и время выполнения в C++:

```

template <int x>
int add(int y) {
  return x+y;
}
  
```

параметр с решёткой
 можно улодобить
 шаблоном параметрам

add<2>(3);

④ - продолжаем фокус.

$$\langle \text{Go } e_0 \rangle$$

↓

$$\langle \text{Rev } e, e_0 \rangle$$

↓ $c \leftarrow e_1$

FI

$$\langle \text{Rev } \#e_1, e_0 \rangle : \#e_1$$

формат говорит о том, что функция после $\#$ расформатируется ничего не возвращает, константа $\#e_1$ - часть и входного, и выходного формата

$$e_0 \rightarrow s.2 e_3$$

$$\langle \text{Rev } s.2 \#e_1, e_3 \rangle : \#e_1$$

конфигурация опасно похожа на родительскую, что делать будем? Составляем

с родительской?

$$s.2 \#e_1 \leftarrow \#e_1$$

$$e_3 \leftarrow e_0$$

не очевидно, как поминать?

А это и есть фокус!

Константа $\#e_1$ в родительской конфигурации соответствует выражению $s.2 \#e_1$

Перенимаем в родительской - добавим штрих

$$\langle \text{Rev } \#e_1', e_0 \rangle : \#e_1'$$

$$\#e_1' \stackrel{\text{def}}{=} s.2 \#e_1$$



$$\langle \text{Rev } s.2 \#e_1, e_3 \rangle$$

$$\rightarrow s.2 \#e_1 : \#e_1$$

Из гипотезы формата следует, что первый аргумент как есть падает в результат, значит, результатом дочерней конфигурации будет

$$\langle \text{Rev } \underbrace{\#e_1'}_{s.2 \#e_1}, \underbrace{e_0}_{e_3} \rangle : \underbrace{\#e_1'}_{s.2 \#e_1}$$

$$\langle \text{Rev } s.2 \#e_1, e_3 \rangle == s.2 \#e_1$$

Результат $s.2 \#e_1 \neq \#e_1$, значит, обобщаем форматную гипотезу

$$s.2 \#e_1 \sqcap \#e_1 = e_2 \#e_1$$

⑤ - продолжаем или завершаем фокус
 $\langle GO\ e.0 \rangle$

\downarrow
 $\langle Rev\ e, e.0 \rangle$

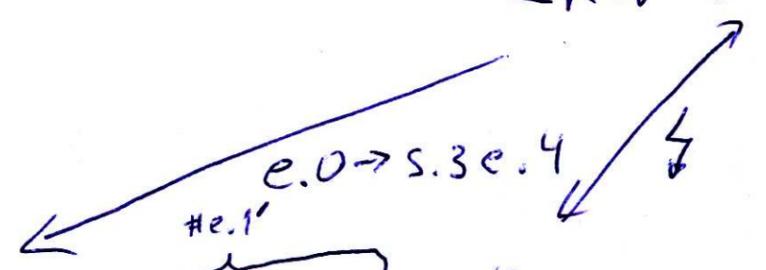
$\downarrow\ c \leftarrow e.1$

F1

$\langle Rev\ \#e.1, e.0 \rangle : e.2 \# e.1$

Формат говорит о том, что функция после расформатирования вернет $e.2$

Еще добавим расформатированную конфигурацию, как $R\ F1$, то её формат будет $\langle F1\ e.0 \rangle : e.2$.



$\langle Rev\ s.3 \# e.1, e.4 \rangle$

Составляем с родительским форматом:

$\langle Rev\ s.3 \# e.1, e.4 \rangle : [\langle Rev\ \#e.1', e.0' \rangle : e.2' \# e.1']$

$\begin{cases} s.3 \# e.1 \leftarrow \#e.1' & \text{результат будет} \\ e.4 \leftarrow e.0' & e.2' s.3 \# e.1, \end{cases}$

составляем с форматом:

$e.2' s.3 \# e.1 : e.2 \# e.1$

Приходим к выводу, что функция возвращает $e.2' s.3 \leftarrow e.2$

~~Рекурсия~~

Рекурсивно вызываем конфигурацию $F2$:

$\langle F1\ e.4 \rangle \leftarrow e.5$

пользуясь форматом гильдой; она вернет

$e.5 \# e.1'$, где

$\#e.1' \equiv s.3 \# e.1$,

т.е. $e.5\ s.3 \# e.1$.

Составляем

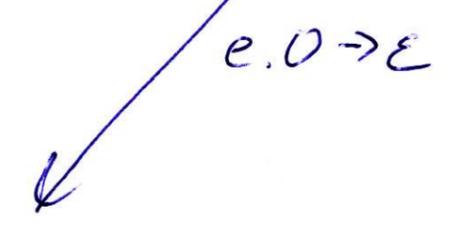
с ожидаемым форматом

$e.5\ s.3 \# e.1 : e.2 \# e.1$,

получается, что как $e.2$

вернется $e.5\ s.3$

или $\langle F1\ e.4 \rangle\ s.3$



$\#e.1 : e.2 \# e.1$,

$e.2 = e$,

т.е. здесь

$\langle F1\ e \rangle$ вернет e

Остаток программы примет вид

$\$ENTRY\ GO\ \{$

$e.0 = \langle F1\ e.0 \rangle\ e;$

$\}$

$F1\ \{$

$s.3\ e.4 = \langle F1\ e.4 \rangle\ s.3;$

$e = e;$

$\}$

конкатенация с e форматна, вымжет из формата $e.2 \# e.1$, где $e \leftarrow e.1$

