

Homework 7
SNU 4190.310, 2015 가을
Kwangkeun Yi
Due: 12/01(Tue), 24:00

Exercise 1 (80pts) “점프마저 설탕”

이번 숙제는 예외상황을 발생시키고 처리하는 설탕을 모두 녹이는 변환기를 만드는 것이다. 현대의 프로그래밍 언어의 많은 장치들은 대부분 설탕인데, 예외상황처리 장치들도 그렇다는 것을 확인해보자.

프로그래밍언어에서 goto/jump/break 등은 실행순서를 변경시키는 명령이고, 이런 명령들을 모두 포섭하면서 그 이상으로 강력한 것이 예외상황처리(exception raise/handling)다.

아래의 언어가 출발어다. 적극적인 계산법(eager evaluation)으로 실행되는 언어다. 이전 숙제에서 다루었던 언어와 유사하다. 다른 점은 예외상황 발생(raise)과 처리(handle) 식이 첨가된 것이다.

$e ::=$	n	natural number
	x	identifier
	$\lambda x.e$	function
	ee	application
	if eee	branch
	$e = e$	number equal check
	raise e	exception raise
	e handle $n e$	exception handling

예외상황처리식 “ e_1 **handle** $n e_2$ ”은 식 e_1 과 거의 같다. e_1 을 계산하면서 상황에 따라 다르게 작동한다.

- e_1 계산이 정상적으로 끝나면 그 결과가 이 예외상황처리식의 결과가 된

다. e_2 는 예외상황이 발생할 경우를 위해 준비해 놓은 식(handler)이다. 예외상황이 발생하지 않았으므로 실행되지 않는다.

- e_1 계산중에 예외상황이 발생하면, 정상 진행을 멈추고 곧바로 현재의 예외상황 처리식으로 꺾음 점프한다. 그리고 준비해 놓은 처리식(handler)이 처리할 수 있는 지 검토후 처리한다.

– 그 예외상황 값이 n 이면 현재의 처리식이 처리할 수 있다. 즉, 처리식(handler) e_2 를 계산하고 그 결과가 현재의 예외상황 처리식의 결과가 된다.

– 그 예외상황 값이 n 이 아니면, 현재의 처리식으로는 그 예외상황을 처리할 수 없다. 이 경우 현재의 예외상황 처리식을 실행 순으로 감싸고 있는 바로 이전의 예외상황 처리식으로 다시 꺾음 점프해서 위의 과정을 반복한다. 이 과정으로 예외상황 처리식을 찾지못하면 프로그램 실행은 갑작스럽게 멈춘다.

예외상황을 발생시키는 식은 “**raise** e ”이다. 우선 e 를 계산한다. 결과는 자연수이어야 한다. 그 자연수 값을 가지고 예외상황을 발생시킨다. 그러면 정상적인 진행이 멈춰지고 이 예외상황을 처리할 처리식을 찾아 꺾음점프해간다.

예를들어, 함수 f 가 다음과 같이 정의되어 있다고 하자.

$$f() = 1 + (E \text{ handle } 99 \ 0) \quad (1)$$

다음과 같은 식을 생각하자

$$f() \text{ handle } 77 \ 10 \quad (2)$$

식 (2)를 실행하자. 함수 $f()$ 가 실행된다. 그러면 함수 내부(식 (1))에서 E 가 실행된다. 그 안에서 예외상황 **raise** 77이 발생했다고 하자. E 를 감쌌던 첫 번째 처리문(식 (1))으로 꺾음 점프한다. 그 처리문은 예외값 99 만을 처리할 수 있을 뿐이다. 따라서 발생한 예외상황은 처리되지 못하고 다음 처리식으로 다시 꺾음 점프한다. 실행순서에서 현재의 처리문을 감싸고 있던 처리식으로(식 (2)). 그 곳에 이르러서야 발생한 예외상황 77이 처리될 수 있다. 그래서 식 (2)의 실행값은 10이 된다.

숙제는, 위의 언어로 작성된 프로그램을 받아서 **raise**와 **handle**이 사라진 식으로 변환하는 함수:

```
removeExn: xexp -> xexp
```

를 정의하는 것이다. 변환된 결과 e 는 e 와 같은 일을 해야 한다. 위 언어의 실행기 `run`으로 다음과 같이 실행시켜서 두 결과가 같아야 한다:

$$\text{run}(e) = \text{run}(\text{removeExn } e)$$

변환할 프로그램은 항상 자연수를 최종적으로 계산하는 프로그램으로 한정한다. 조교는 위 언어의 실행기 `run`과 `mexp`의 파서를 제공할 것이다.

```
type xexp = Num of int
          | Var of string
          | Fn of string * xexp
          | App of xexp * xexp
          | If of xexp * xexp * xexp
          | Equal of xexp * xexp
          | Raise of xexp
          | Handle of xexp * int * xexp
```

□