Homework 5
COSE312, Spring 2020

Hakjoo Oh

Due: 6/3, 6/19 23:59

Problem 1 이번 숙제에서는 수업시간에 다룬 `S`를 `T`로 변환하는 컴파일러를 구현해 봅시다. 출발 언어(source language) `S`를 다음과 같이 정의했습니다.

$$
\begin{array}{rcll}
\textit{program} & \to & \textit{block} & \\
\textit{block} & \to & \textit{decls}\ \textit{stmts} & \\
\textit{decls} & \to & \textit{decls}\ \textit{decl} \mid \epsilon & \\
\textit{decl} & \to & \textit{type}\ x & \\
\textit{type} & \to & \texttt{int} \mid \texttt{int[}n\texttt{]} & \\
\textit{stmts} & \to & \textit{stmts}\ \textit{stmt} \mid \epsilon & \\
 & & & \\
\textit{stmt} & \to & \textit{lv}\ \texttt{=}\ e & \\
 & \mid & \texttt{if}\ e\ \textit{stmt}\ \textit{stmt} & \\
 & \mid & \texttt{while}\ e\ \textit{stmt} & \\
 & \mid & \texttt{do}\ \textit{stmt}\ \texttt{while}\ e & \\
 & \mid & \texttt{read}\ x & \\
 & \mid & \texttt{print}\ e & \\
 & \mid & \textit{block} & \\
 & & & \\
\textit{lv} & \to & x \mid x\texttt{[}e\texttt{]} & \\
 & & & \\
e & \to & n & \text{integer} \\
 & \mid & \textit{lv} & \text{l-value} \\
 & \mid & e\texttt{+}e \mid e\texttt{-}e \mid e\texttt{*}e \mid e\texttt{/}e \mid \texttt{-}e & \text{airthmetic operation} \\
 & \mid & e\texttt{==}e \mid e\texttt{<}e \mid e\texttt{<=}e \mid e\texttt{>}e \mid e\texttt{>=}e & \text{conditional operation} \\
 & \mid & \texttt{!}e \mid e\texttt{||}e \mid e\texttt{\&\&}e & \text{boolean operation}
\end{array}
$$

OCaml 자료형으로는 아래와 같이 정의됩니다.

```
type program = block
and block = decls * stmts
and decls = decl list
```

```
and decl  = typ * id
and typ   = TINT | TARR of int
and stmts = stmt list
and id    = string
and stmt  = ASSIGN of lv * exp
          | IF of exp * stmt * stmt
          | WHILE of exp * stmt
          | DOWHILE of stmt * exp
          | READ of id
          | PRINT of exp
          | BLOCK of block
and lv   = ID of id | ARR of id * exp
and exp  =  NUM of int
          | LV of lv | ADD of exp * exp
          | SUB of exp * exp | MUL of exp * exp
          | DIV of exp * exp | MINUS of exp
          | NOT of exp | LT of exp * exp
          | LE of exp * exp | GT of exp * exp
          | GE of exp * exp | EQ of exp * exp
          | AND of exp * exp | OR  of exp * exp
```

도착 언어(target language) `T`는 다음과 같이 정의했습니다.

$$
\begin{array}{rcl}
\textit{program} & \rightarrow & \textit{LabeledInstruction}^* \\
\textit{LabeledInstruction} & \rightarrow & \textit{Label} \times \textit{Instruction} \\
\textit{Instruction} & \rightarrow & \texttt{skip} \\
 & | & x = \texttt{alloc}(n) \\
 & | & x = y \ \textit{bop} \ z \\
 & | & x = y \ \textit{bop} \ n \\
 & | & x = \textit{uop} \ y \\
 & | & x = y \\
 & | & x = n \\
 & | & \texttt{goto} \ L \\
 & | & \texttt{if} \ x \ \texttt{goto} \ L \\
 & | & \texttt{ifFalse} \ x \ \texttt{goto} \ L \\
 & | & x = y[i] \\
 & | & x[i] = y \\
 & | & \texttt{read} \ x \\
 & | & \texttt{write} \ x \\
\textit{bop} & \rightarrow & \texttt{+} \mid \texttt{-} \mid \texttt{*} \mid \texttt{/} \mid \texttt{>} \mid \texttt{>=} \mid \texttt{<} \mid \texttt{<=} \mid \texttt{==} \mid \texttt{\&\&} \mid \texttt{||} \\
\textit{uop} & \rightarrow & \texttt{-} \mid \texttt{!}
\end{array}
$$

OCaml 자료형으로 다음과 같이 정의됩니다.

```
type program = linstr list
and linstr = label * instr
and instr =
  | SKIP
  | ALLOC of var * int  (* x = alloc(n) *)
  | ASSIGNV of var * bop * var * var (* x = y bop z *)
  | ASSIGNC of var * bop * var * int (* x = y bop n *)
  | ASSIGNU of var * uop * var  (* x = uop y *)
  | COPY of var * var           (* x = y *)
  | COPYC of var * int          (* x = n *)
  | UJUMP of label              (* goto L *)
  | CJUMP of var * label        (* if x goto L *)
  | CJUMPF of var * label       (* ifFalse x goto L *)
  | LOAD of var * arr           (* x = a[i] *)
  | STORE of arr * var          (* a[i] = x *)
  | READ of var                 (* read x *)
  | WRITE of var                (* write x *)
  | HALT
and var = string
and label = int
and arr = var * var
and bop = ADD | SUB | MUL | DIV | LT | LE | GT | GE | EQ | AND | OR
and uop = MINUS | NOT
```

1. (100pts, **due: 6/3 23:59**) 주어진 S 프로그램을 동일한 의미를 가지는 T 프로그램으로 변환하는 함수 translate를 작성하세요.

   ```
   translate : S.program -> T.program
   ```

   저장소[1]의 translator.ml에 있는 위 함수를 완성하면 됩니다.

2. (100pts, **due: 6/19 23:59**) 변환된 T 프로그램을 최적화하는 함수 optimize를 구현하세요. 최적화 전과 후의 프로그램은 의미가 같아야 합니다.

   ```
   optimize: T.program -> T.program
   ```

   optimizer.ml에 있는 위 함수를 완성하세요.

예를 들어, 아래와 같이 실행할 때

[1] https://github.com/kupl/Compilers2020/tree/master/hw5

```
$ Make
$ ./run test/t0.s
```

다음과 같은 결과를 얻어야 합니다.

```
== source program ==
{
 int x;
 x = 0;
 print (x+1);
}
== execute the source program ==
1
== translated target program ==
0 : x = 0
0 : t4 = 0
0 : x = t4
0 : t1 = x
0 : t2 = 1
0 : t3 = t1 + t2
0 : write t3
0 : HALT
== execution of the translated program ==
1
The number of instructions executed : 7
== optimized target program  ==
0 : t3 = 1
0 : write t3
0 : HALT
== execution the optimized target program ==
1
The number of instructions executed : 2
```