Principles of Programming Languages

Fall, 2022

Programming Assignment #2

< 문　　제 **>**

※ 입력된 프로그램이 제시된 문법 (Grammar)에 맞는지 판단하고, 해석 및 실행하는 인터프리터를 구현하시오.

< 문　　법 **>**

<start> → <functions>

<functions> → <function> | <function> <functions>

<function> → <identifier> { <function_body> }

<function_body> → <var_definitions> <statements> | <statements>

<var_definitions> → <var_definition> | <var_definition> <var_definitions>

<var_definition> → variable <var_list> ;

<var_list> → <identifier> | <identifier>, <var_list>

<statements> → <statement> | <statement> <statements>

<statement> → call <identifier>; | print_ari; | <identifier>;

<identifier> → any names conforming to the C identifier standard

< 세부사항 **>**

※ 개발 조건

① C/C++/C#/Java/Python으로 개발된 **Command Line Application**만 허용
(오픈소스기반 GCC, Python 추천)

② 자신의 학번으로 명명된 폴더 내에 작성된 모든 소스 코드 저장
③ Java의 경우, 폴더 최상단에 Main.jar를 Runnable Jar 파일로 생성한 다음, 'java –jar Main.jar'를 실행
④ Python의 경우, main.py을 진입점(entry point)으로 하고 'python main.py'를 실행

※ 주의**!:** 프로그램을 평가할때, 컴파일이 되지 않으면 채점되지 않으니, 꼭 외부문서(External Document)에 제출하는 프로그램을 프로그램 수행에 필요한 요소와 컴파일부터 수행시키는 방법까지 잘 설명 해주셔야 합니다.

※ 입력**:** 명령행 옵션으로 입력 파일 지정 (Command Line 파라미터로 파일명이 주어짐. (예: "python main.py **file_name.txt**" or "program **file_name.txt**")

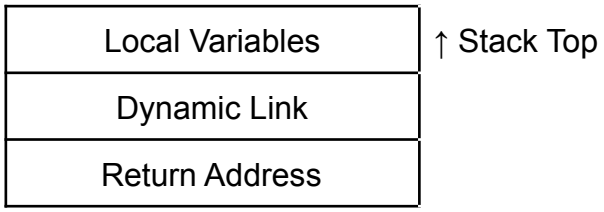
※ 출력**:** 주어진 문법에 따라 입력 파일의 프로그램 문장을 분석한다. 분석 결과, 문법에 맞지 않으면, "Syntax Error."라고 출력한 후 종료한다. 문법에 맞으면, "Syntax O.K."를 출력하고, 프로그램에 명시된 기능을 수행한다.

※ 처리 조건

(1) 이 프로그램에서 사용하는 언어는, Dynamic Scope Language이며, non-local 변수의 참조는 Deep Access 방식 (교재, 영문판, Chapter 10, pp. 461 ~ 463 참고)을 이용하는 것으로 가정한다.
(2) 프로그램은 구문 분석 및 실행의 단계로, 2단계로 동작한다.
(3) 모든 변수명과 함수명의 길이에는 제한이 없다.
(4) variable, call, print_ari는 모두 예약어 (reserved word)이다.
(5) variable은 함수의 지역변수를 정의한다.
(6) call문은, 다른 함수를 호출하는 명령어이다. 프로그램의 구문 분석 단계에서는, call문 뒤의 <identifier>가 정의된 함수명인지 여부를 확인하지 않으며, 실제 실행단계에서, 정의되지 않은 함수명인 경우, 오류 메시지 ("Call to undefined function: function_name")을 출력하고, 종료한다 (단, function_name은 정의되지 않은 함수명임). 정의된 함수에 대한 올바른 호출인 경우, 호출된 함수의 ARI (Activation Record Instance, Chapter 10)를 런타임-스택에 추가한다.
(7) main이라는 이름을 갖는 함수는 반드시 한 개 존재하여야 하며, 프로그램 실행은 main 함수로 부터 시작한다. 실행 단계에서 main 함수가 정의되지

않은 경우, 프로그램은 오류 메시지 (“No starting function.”)을 출력하고 종료한다.

(8) AR (Activation Record)의 구조는 다음과 같이 복귀주소 (Return Address), 동적 링크 (Dynamic Link), 그리고 지역변수 (Local Variables)들로 구성된다.



(9) print_ari는 현재의 런타임 스택의 모든 내용을 화면에 출력하는 명령어로서 예약어이다.

(10) ARI 내 각 항목의 크기는 1 word이며, 복귀주소와 동적 링크의 local_offset은 각각 0, 1이다. 또한, 지역변수의 local_offset은 2부터 시작한다.

(11) 어떤 함수 내에서, n번째 실행문의 주소는 (function_name: n - 1)로 표시된다. 단, 지역변수 선언문은 실행문에 포함되지 않는다.

(12) 복귀주소는 call 명령 다음의 주소를 가리킨다.

(13) 함수 Q가 P를 호출할 때, P의 ARI의 동적 링크는 Q의 ARI의 bottom을 가리킨다.

(14) 실행문에서 <identifier>가 나타나면, 변수 참조를 의미한다. 변수 참조 시에는 해당 변수의 link_count와 local_offset 값을 출력한다. 여기서 link_count는 동적 링크를 따라가는 (Traverse) 횟수를 의미한다.

(15) 하나의 이름이 변수명과 함수명으로 동시에 사용될 수 없다. 만일 동시에 사용된 경우, 오류 메시지 (“Duplicate declaration of the identifier or the function name: identifier/function_name”)을 출력하고 실행을 종료한다. (단, identifier/function_name은 중복 정의된 변수명 또는 함수명임)

(16) 같은 이름을 갖는 함수가 2개 이상 정의될 수 없다. 만일 2개 이상 정의된 경우, 오류 메시지 (“Duplicate declaration of the function name: function_name”)을 출력하고 실행을 종료한다. (단, function_name은 중복 정의된 함수명임)

(17) 하나의 함수 안에서 같은 이름을 갖는 지역변수가 2개 이상 정의될 수 없다. 만일 2개 이상 정의된 경우, 오류 메시지 (“Duplicate declaration of the

identifier: identifier_name")을 출력한 다음, 나중에 중복 정의된 변수명을 삭제한 후, 실행을 계속한다. (단, identifier_name은 중복 정의된 변수명 임)

(18) 입력 파일에서 ASCII 코드 32이하인 모든 문자는 white-space로 간주되며, white-space는 각 token을 구별하는 용도 이외에는 모두 무시된다.

(19) 기타 구현 시 요구되는 세부 사항은 직접 결정하고, internal/external document에 그 이유를 상세히 기술한다. (internal: 어느 부분에 그 기타 구현을 처리하는지 표기, external: 세부사항에 대한 이유 및 설명 기술)

< 처리 예 >

※ 입력 파일

```
first {
        variable a, b, c;
        call second;
        p;
}
second {
        variable x, a;
        b;
        print_ari;
        q;
}
main {
        variable p, q;
        call first;
}
```

※ 실행 결과

```
Syntax O.K.

second:b => 1, 3

second:Local variable: a
```

Local variable: x
Dynamic Link: 2

first: Local variable: c
Local variable: b
Local variable: a
Dynamic Link: 0
Return Address: main: 1

main: Local variable: q
Local variable: p

second:q => 2, 1

first:p => 1, 0

< 제출 관련 사항 **>**

제출물

- Internal / External Documents
  - Internal Document: 작성한 소스코드를 간략히 설명 (어느 부분에서 무엇을 작업하는 지 정도 요약, 소스코드 내 작성)
  - External Document: 프로그램의 실행 방법 및 요구사항 자세히 설명바람 (이 문서를 보고 제출한 소스코드를 컴파일하고, 실행방법을 따라서 수행할 예정, 문서로 작성 후**, pdf** 파일로 제출)
- 프로그램 소스 코드

제출 방법

- 모든 제출물을 압축하여 학번_이름**.zip** 형식으로 eclass에 제출

제출 마감 일시

- **2022**년 **12**월 **23**일 (금) 오후 **11:59** (Late Penalty: -5% / Day)