HomeWork1

Hyunki Lee

Panther# 002-34-4677

1.

<S> is starting statement.

Rule <A> can be represented a single a or more a's

Rule <B> can be represented a single b or more b's

Rule <C> can be represented a single c or more c's

The grammar is that <A><B><C> is given, and we can replace nonterminal values to terminal values. One or more a's for <A>, one or more b's for<B>, one or more c's for<C>.

2.

Answer: a, d

Start with <A> a <B> b

<A> can be <A> b or b

<B> can be a <B> or a

<A> can be one or more b's, <B> can be one or more a's.

The grammar always ends with 'b'. Therefore, choice c cannot be answer.

We must have at least 2 a's because we have 'a' right before <B>. <B> should have at least one

a. Thus, choice b is not an answer.

For a.) <A> a <B> b ➔ b a <B> b ➔ b a a b

For d.) <A> a <B> b ➔ b <A> a <B> b ➔ b b a <B> b ➔ b b a a b

3. Four criteria, “While B do S end”, correctness of a logical pretest loop construct

P = { power =1; i=1;} precondition

B = {I <=n}

Q ={ power = x^n} Post condition

S=[

Power = power *x;

i = i+1;

]

Four criteria

1) P => I : The invariant is initially true
2) {I and B} S {I} : Each execution of the loop preserves the invariant
3) (I and(not B)) => Q : The invariant and the loop exit condition imply the postcondition.
4) The loop terminates

First criteria: invariant = power. The power is initialized 1 (power =1;), thus power is greater than 0. i is also initialized as 1 (i=1;), thus i is greater than 0. Satisfied the first criteria.

Second criteria: after passing the loop the power and i keep their value greater than 0. Therefore, the second criteria is true. (if x is greater than 0).

Third criteria: when B get false that is ‘i > n’, value of i is still true. Q is ‘power = x^n, and it is true.

Forth criteria: when ‘i>n’ , the loop terminates.

4. operational semantic definition

a. Java do-while

Syntax:

```
do{
statements;
} while(expression);
```

Operational semantic definition

```
Loop:
Statements;
if expression == false goto Out
goto Loop
Done:
```

b. C++ if-then-else

Syntex:

```
If(Boolean_expression){

Statements_1;}

else{

statements_2;}
```

Operational semantic definition

```
If(Boolean_expression ==ture) goto L1

goto L2

L1: Statements_1;

L2: Statements_2;
```

5. Write denotational semantics mapping function.

a. Java for

Syntax

```
for(variable initialization; condition; change variable value){

        statement();

}
```

Denotational semantics mapping function

```
$M_{for}$(for ($E_1$;$E_2$;$E_3$){Ls},s)=
    if $E_1$ is not empty
        If $M_{Expr}(E_1,s)$ ==error return error
    if $E_2$ is not empty
        If $M_{BoolExpr}(E_2,s)$ ==error return error
        else if $M_{BoolExpr}(E_2,s)$ == false return s
        else if $M_{StatementList}(Ls,s)$ == error return error
        else if $E_3$ is not empty
            If $M_{Expr}(E_3,s)$ ==error return error
$M_{forloop}$((for (E1;E2;E3){Ls},$M_{statementList}(Ls;E_3)$))
```

b. Java do-while

Sysntax

```
Do{
    Statements;
}while(condition);
```

Denotational semantics mapping function

```
$M_{dowhile}$ (do Ls while Boolean, s) =
    if $M_{boolean}$ (Boolean,s) == error return error
        else if $M_{boolean}$(Boolean,s) == false return statement
            else if $M_{Ls}$(Ls,s) == error return error
                else $M_{dowhile}$(do Ls while Boolean, $M_{Ls}$(Ls,s))
```

c. C switch

Syntax

```
switch(expression){
    case constant_expression:
        statements;
        default;
    }
```

Denotational semantics mapping function

```
$M_{switch}$(<expression><switch block>,s) =
    if M(<expression>,s) == error then return error
        else M(<switch block>)
```

8. How many lexemes?

Answer: 82

I break down each lexeme

```
        public /class/ CountDigits/ { /                                  -> 4
                public /static/ void/ main/(/String/[/] /args/) /{/      -> 11
                        SimpleIO/./prompt/(/"Enter an integer: "/)/;/     -> 7
                        String /userInput/ = /SimpleIO/./readLine/(/)/;/   -> 9
                        Int/ number/ = /Integer/./parseInt/(/userInput/)/;/   -> 10
                        Int/ numDigits/ =/ 0/;/                              -> 5
                        While/ (/number/ >/ 0/)/ {/                          -> 7
                                Number/ //=/ 10/;/                           -> 5
                                numDigits/+/+/;/                             -> 4
                        /}                                                   ->1
                System/./out/./println/(/"The number "/ +/ userInput /+ /" has "/ +/
                numDigits /+/ " digits"/)/;/                                         ->17
                }                                                             ->1
}                                                                              -> 1
```

Total: 82

9. Find errors

1) private int temperature → syntax error. need semicolon ➔ private int temperature;

2) temperature = 0.0; ➔ semantic error. temperature is int and 0.0 is double ➔ temperature = 0;

3) temperature =+ degrees; ➔ lexical error. It should be += ➔ temperature += degrees;

4) public getTemperature(){ ➔ syntax error. We must define return type ➔ public int getTemperature(){

5) public string tostring(){ ➔ Syntax error. string must be String ➔ public String tostring(){

6) return temperature + degrees; ➔ semantic error. temperature is int, but we need to return string. Also, degrees are not defined.

10.

```
#include <stdio.h>
#include<string.h>
#define MAX 1000
int main(){

char c[MAX];
FILE *fp;
fp = fopen("Q10.txt", "r");
int i;
int size;

    if ( fp == NULL) {
    printf("Cannot open file");
    return -1;
    }

fscanf(fp, "%[^\n]", c);
size = strlen(c);

    for(i=0; i<size; i++){
        if((c[i]>='0'&& c[i]<='9')||(c[i]>='a'&& c[i]<='z')||(c[i]>='A'&& c[i]<='Z')){

            printf("%c",c[i]);
        }
        else if(c[i] == ' '){
            printf("\n");
        }
        else{
            printf("\n");
            printf("%c\n",c[i]);
        }
    }

fclose(fp);
return 0;

}
```

Repl.it link:

https://repl.it/@todok4636/PLCQ10HyunkiLee#main.c