BITS Pilani, Hyderabad Campus
Department of Computer Science and Information Systems
Second Semester, 2024-25
CS F363 Compiler Construction
Project part 2 (12 Marks)

The description of the toy language is the same as given in Part 1.

1 Tasks

1. *[5 marks]* Write a Bison program to check if the given program is syntactically correct for the toy language.
2. *[7 marks]* Extend your Bison program to print an abstract syntax tree (AST) for the given input program if it is syntactically correct. Print your AST as a generalized list (see Lab sheet 9 for more details).

2 Input and output format

The input program will be given in a text file, whose name will be given at the runtime. Use the following code for the same purpose.

```
int main(int argc, char *argv[]) {
    if (argc != 2) {
        fprintf(stderr, "Usage: %s <input file>\n", argv[0]);
        return 1;
    }

    yyin = fopen(argv[1], "r");
    if (!yyin) {
        perror("Error opening file");
        return 1;
    }

    yyparse();
    fclose(yyin);
    return 0;
}
```

The output should be displayed on the screen.

3 Submission guidelines

1. You work on the assignment with your group members and are strongly discouraged from discussing it with others.

2. You can refer to the Internet/Web resources only to understand the syntax of the Bison/Flex tools.

3. Taking code from the Internet/Web is strictly prohibited. If you do, it will lead to severe penalties.

4. Create a zip file with the following folders/files in it.

   (a) Create separate folders for Syntax Analysis and AST. Keep the respective files and all necessary files in the folders.

   (b) Create a `make` file to compile bison and flex codes and to generate an executable file (let it generate a default file `a.out`). Keep the `make` file in both folders: Syntax Analysis and AST.

   (c) A PDF file that contains the information of the group members (BITS ID, name, and email ID).

5. Due date is 15 April 2025 at 11:59 PM. Late submissions will be allowed up to 24 hours after the deadline with a penalty of `2%` per hour.

4 Test Cases

Syntax Analysis

```
//Input:

begin program:
begin VarDecl:
(a, int);
(b, int);
end VarDecl
a := (5, 10);
b := (7, 8);
end program


//Output: Successfully parsed !!!
```

```
//Input:

begin program:
begin VarDecl:
(x, int);
(y, int);
(z, int);
(binaryVal, int); //invalid variable name. This is an error.
end VarDecl
x := (10, 10);
y := (11, 10);
z := x + y;
print("Sum = @", z);

binaryVal := (1010, 2);
print("Binary value = @", binaryVal);
end program

//Output: Syntax error !!!
```

```
//Input:

begin program:
begin VarDecl:
(a, int);
(b, int);
end VarDecl
a := 77;   //this line has a syntax error that 77 is not a valid number format.
b := (1001, 2);
print("a = @", a);
print("b = @", b);
end program


//Output: Syntax error !!!
```

```
//Input:

begin program:
begin VarDecl:
(x, int);
end VarDecl
x := (1, 10);
if (x > (0, 10))
x := (2, 10);   // compound block is missing
end program


//Output: Syntax error !!!
```

```
//Input:

begin program:
begin VarDecl:
(i, int);
end VarDecl
i := (5, 10);
if (i < (10, 10))
begin
i := (1, 10);
print("@", i);
end   //this line has error semicolon(;) is missing after end
end program


//Output: Syntax error !!!
```

```
//Input:

begin program:
begin VarDecl:
(i, int);
end VarDecl
i := (5, 10);
if (i > (10, 10))
begin
i := (0, 10);
end;  // semicolon (;) is placed extra; this is an error
else
begin
i := (20, 10);
end;
end program

//Output: Syntax error !!!
```

```
//Input:

begin program:
begin VarDecl:
(number, int);
(sum, int);
(a, int);
(b, int);
(arr[6], char);
end VarDecl
number := (3, 10);
sum := (0, 10);
while (number > (0, 10)) do
begin
sum := sum + number;
number := number - (1, 10);
end;

b := (20, 10);
for a := (10, 8) to b + (10, 2) inc (1, 10) do
begin
print("The value of a = @", a);
end;

end program

//Output: Successfully parsed !!!
```

```
//Input:

begin program:
begin VarDecl:
(a, int);
(b, int);
(x, int)    //semicolon (;) is missing. This is an error
end VarDecl
a := (10, 10);
b := (5, 10);

x := (3, 10);
while (x > (0, 10)) do
begin
print("@", x);
x := x - (1, 10);
end  //semicolon (;) is missing. This is an error

for a := a + (3, 10) to (1, 10) dec b - (2, 10) do
begin
print("a = @", a);
end;

end program

//Output: Syntax error !!!
```

AST

The AST is not unique; your version might differ from the one shown.

Input: 1

```
begin program:
begin VarDecl:
(x, int);
(y, int);
(z, int);
(binaryval, int);
end VarDecl
x := (10, 10);
y := (11, 10);
z := x + y;
print("Sum = @", z);

binaryval := (1010, 2);
print("Binary value = @", binaryval);

end program
```

AST in the form of generalized list:

```
(
  (x int)
  (
    (y int)
    (
      (z int)
      (
        (binaryVal int)
        (
          (:= x (10 10))
          (
            (:= y (11 10))
            (
              (:= z (+ x y))
              (
                (print "Sum = @" z)
                (
                  (:= binaryVal (1010 2))
                  (print "Binary value = @" binaryval)
                )
              )
            )
          )
        )
      )
    )
  )
)
```

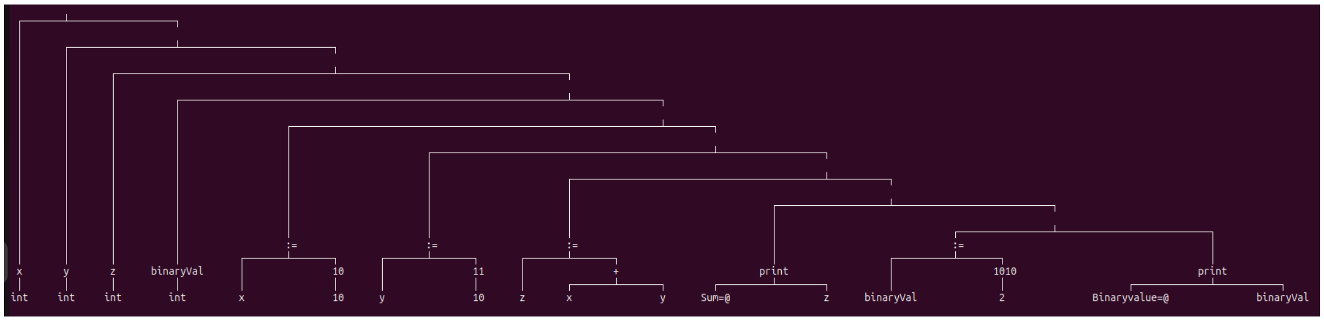

Figure 1: Tree structure of the AST

Input: 2

```
begin program:
begin VarDecl:
(i, int);
end VarDecl
i := (5, 10);
if (i > (10, 10))
begin
i := i + (0, 10);
i := i + i;
end
else
begin
i := (20, 10);
end;
end program
```

AST in the generalized list form

```
(
  (i int)
  (
    (:= i (5 10))
    (
      (if
        (> i (10 10))
        (
          (:= i (+ i (0 10)))
          (
            (:= i (+ i i))
          )
        )
        (
          (:= i (20 10))
        )
      )
    )
  )
)
```

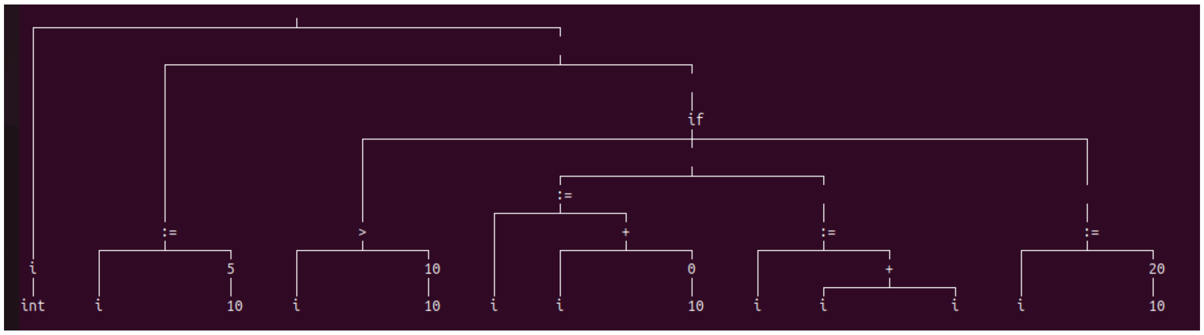

Figure 2: AST for the above test case

Input: 3

```
begin program:
begin VarDecl:
(a, int);
(b, int);
end VarDecl
a := (10, 10);
b := (5, 10);
for a := a + (3, 10) to (1, 10) dec b - (2, 10) do
begin
print("a = @", a);
end;
end program
```

AST in the generalized list form

```
(
  (a int)
  (
    (b int)
    (
      (:= a (10 10))
      (
        (:= b (5 10))
        (
          (for
            (:= a (+ a (3 10)))
            (1 10)
            (dec (- b (2 10)))
            (
              (print "a = @" a)
            )
          )
        )
      )
    )
  )
)
```

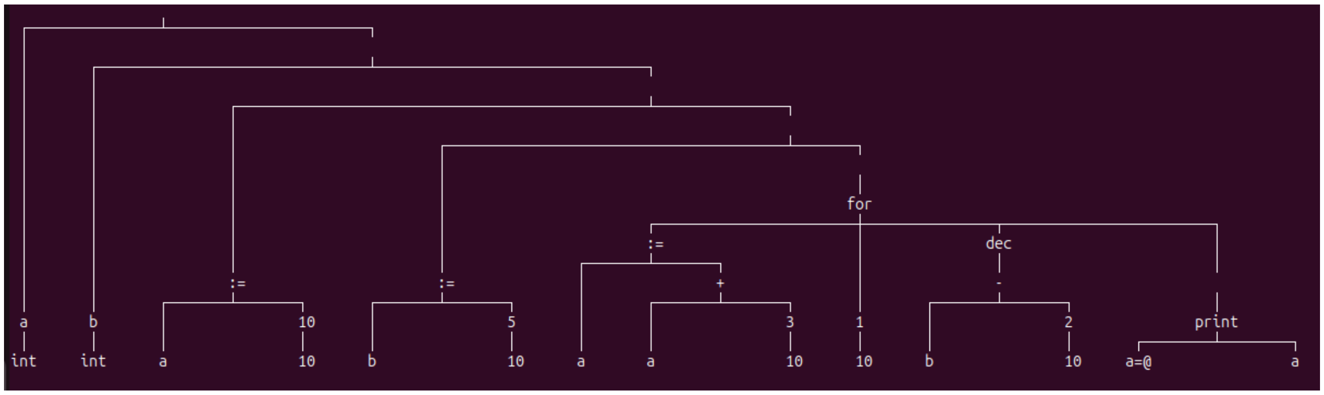

Figure 3: AST for the above test case