Evaluation Standard

July 28, 2022

Name: email: Student id:

You may use and modify the test cases provided in the course web page. You may make up new test cases fitting your compiler. You can also introduce new features and related test cases into your compiler.

You will need to implement three built-in routines:

```
printInt( a+15 );
printReal( b+c/3.14 );
printString( "The size of the graph is" );
```

The three built-in routines can be implemented in C or RISC-V programs (if the target platform is RISC-V). We will provide you with sample routines.

In this project, we will use the RISC-V platform for code generation. If you generate code for other platforms, please tell the instructor before you implement your compiler. There might be extra credits for other platforms.

Each item below will earn 1 point. A scaling formula will be used to translate your total points to a numerical score.

1. (part 1) Filter out comments. (no point for this item)
2. (part 1) Use the longest-match rule in scanning. (no point for this item)
3. (part 1) Scientific notation.
4. (part 1) Signed integers and floats.
5. (part 3) Check for duplicated declarations.

6. Addition and subtraction operations (for integer type).

7. Addition and subtraction operations (for real type).

8. Multiplication and division operations (for integer type).

9. Multiplication and division operations (for real type).

10. Comparision operations, ==, >, >=, <, <=, ! = (for integer type).

11. Comparision operations (for real type).

12. Arithmetic operations (addition, subtraction, multiplication, division, comparison) for variables of one-dimension arrays, such as
    ```
    array [ x+3 ..  y*4 ] of integer
    array [ x+3 ..  y*4 ] of real
    ```

13. Arithmetic operations (addition, subtraction, multiplication, division, comparison) for variables of multi-dimension arrays, such as
    ```
    array [ x+3 ..  y*4 ] of array [ z+6 ..  w/7 ] of integer
    array [ x+3 ..  y*4 ] of array [ z+6 ..  w/7 ] of real
    ```

14. Evaluate constant expressions, such as `2+5*4/2`, in the compiler.

15. Assignment statements of simple types (integer and real).

16. Assignment of 1-dimensional arrays.

17. Assignment of multi-dimensional arrays.

18. Check out-of-bound array indices at compile time.

19. Generate code to check out-of-bound array indices at run time.

20. Simple if statements.

21. Simple while loops.

22. Nested if statements.

23. Nested while loops.

24. Arbitrary combination of assignment statements, if statements, and while loops.

25. Subprograms (procedures and functions, including recursive ones) without parameters and without return values.

26. Subprograms with (integer and real) parameters and with (integer and real) return values (including recursive ones).

27. Subprograms with array parameters and with (integer and real) return values (including recursive ones).

28. PrintInt, PrintReal, and PrintString.

29. There could be constant folding of various degrees of difficulty. For example, `a = 3 + 4 + b; a = 3 + b - 4; a = 3 + b - 4 + c + 5;`

Possible extensions include

1. add structure/record and related operations

2. add static variables similar to C

3. add modular structure similar to Modula-2 or Java Classes.

4. add && and || logical operators

5. add for-loops

6. add repeat-until loops

7. . Check if a variable is initialized. (You will need to draw the control flor diagram for this item.)

8. Add overload resolution for Minipascal.

9. others

–