Compilers Project
A3E

Table of Contents

Introduction

- A3E is a simple programming language compiler similar to C++.
- A3E is handling the most common datatypes like integers, floats, bools, and strings.
- Also, A3E is supporting variables, constants, mathematical and logical expressions, conditional if, while loops, for loops, repeat until, and switch case.

Used Tools and Technologies

Tool or Technology	Description
Flex	Fast Scanner Generator like Lex
Bison	Parser Generator like Yacc

Tokens

1. Variables and constants declaration

- Define variable: *datatype id = value;*
    i. Example: *int a = 5;*
- Define constant: *const datatype id = value;*
    i. Example: *const int a = 5;*

2. Mathematical and Logical expressions

- Mathematical operations
    i. +, -, *, /, %
- Logical operations
    i. ||, &&, ^, !
    ii. >, <, >=, <=
- Any level of parentheses/complexity.

3. Assignment statements

- Variable = expression
- Example: *a = 5 * b + c;*

4. If-endif, if-else statements

```
if (a < 5) {
    a = a + 1;
} endif;

if (a < 5) {
    a = a + 1;
} else {
    a = a + 5;
}

```

5. While loops

```
while (/* expr */) {
    /* body */
}

```

6. For loops

```
for (/* header; expr; expr */) {
    /* body */
}

```

7. Repeat until

```
repeat {
    /* body */
} until (/* expr */);

```

8. Switch case

```
switch (/* var */) {
    case 1:
        /* code */
        break;
    default:
        /* code */
        break;
}
```

9. Block structures

```
{
    /* code */
    {
        /* code */
    }
}
```

Quadruples

Quadruple	Description
ADD s1, s2, R	Pop the top 2 values of the stack (s1, s2) and push s1 + s2
SUB s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push s1 - s2
MUL s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push s1 * s2
DIV s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push s1 / s2
MOD s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push s1 % s2
LT s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push true if s1 < s2 else false
GT s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push true if s1 > s2 else false
LE s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push true if s1 <= s2 else false
GE s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push true if s1 >= s2 else false
EQ s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push true if s1 == s2 else false
NE s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push true if s1 != s2 else false
NOT s1, r	Pop the top 2 values of the stack (s) and push !s
AND s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push true if s1 && s2
OR s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push true if s1 || s2
XOR s1, s2, r	Pop the top 2 values of the stack (s1, s2) and push s1 ^ s2
PUSH value	Push value to the stack
POP dst	Pop the top of the stack in into dst
L<num>:	Add label with number num
JMP L	Unconditional jump to L
JZ L	Jump to L if stack top == 0
B<num>:	Add break label with the scope number