# Introduction to exercise 4a: Symbol table

CS-460: Programming Language

Robert Bruce

# What is a symbol table?

A symbol table is typically a linear linked-list data structure which stores the following information:

- The names of all functions, procedures, globally-defined variables, and locally-defined variables within an input program.
- Run-time values for each variable within an input program during program execution (i.e. interpreting the input program).
- The scope of variables within an input program.

# How is a symbol table useful?

**Symbol table is very useful for:**

- Locating the names of all procedures and functions within an abstract syntax tree. For example, a given entry in a symbol table could point to the node of an Abstract Syntax Tree (AST) where that procedure or function is defined. This is useful when running the interpreter.
- Managing variable values during run-time of an interpreted program.
- Determining undefined variables, procedures, or functions.
- Determining ambiguous variable declarations within a given scope.

# Example symbol table

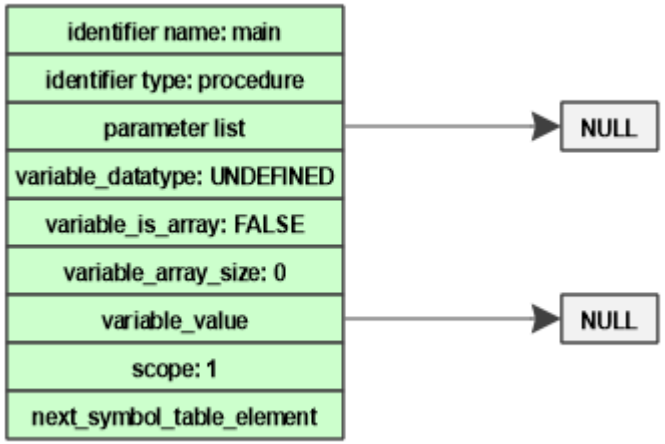
```
procedure main (void)
{
  int counter;
  int state;

  counter = 1;
  while (counter <= 100)
  {
    state = 0;
    if ((counter % 3) == 0)
    {
      state = 1;
    }
    if ((counter % 5) == 0)
    {
      state = state * 2 + 2;
    }
    if (state == 1)
    {
      printf ("Fizz");
    }
    else
    {
      if (state == 2)
      {
        printf ("Buzz");
      }
      else
      {
        if (state == 4)
        {
          printf ("Fizzbuzz");
        }
        else
        {
          printf ("%d", counter);
        }
      }
    }
    counter = counter + 1;
    if (counter <= 100)
    {
      printf (", ");
    }
    else
    {
      printf ("\n");
    }
  }
}
```

# Example symbol table

**Example input program:**

```
procedure main (void)
```

**Graphically, the corresponding entry in our symbol table for the input above might contain the following attributes:**

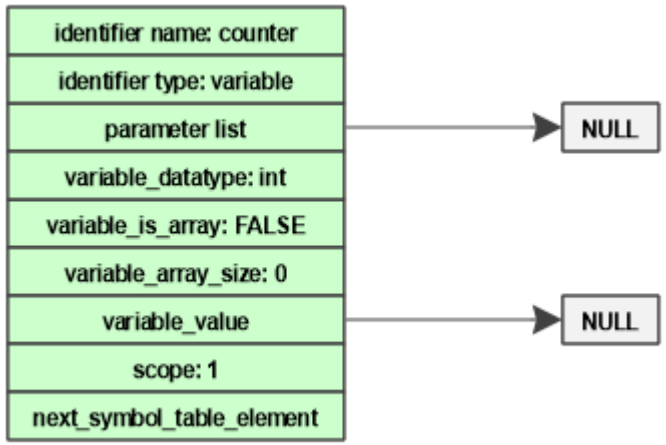


# Example symbol table

**Example input program:**

```
int counter;
```

**Graphically, the corresponding entry in our symbol table for the input above might contain the following attributes:**

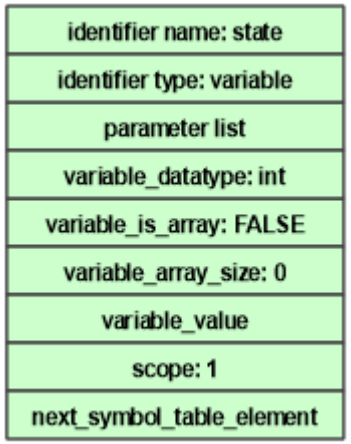


# Example symbol table

**Example input program:**

```
int state;
```

**Graphically, the corresponding entry in our symbol table for the input above might contain the following attributes:**



# Example symbol table