SaanpCD

SaanpCD is a programming language developed by four Computer Science undergraduates as part of their Compilers course (`CS F363`).

This language is a C-like language with a few extra features `import`ed from Python.

Features of the language

- The language is not a loosely-typed language, data types exist. The data types available are:
  - `int` (Integer)
  - `float` (Floating-point)
  - `char` (8-bit ASCII value)
  - `string` (Collection of `char`s)
  - `bool` (Boolean literal)
- Procedures (blocks of code that are repeated and can be called by the programmer) and functions (procedures that return a value to a variable) are supported. The following are features of procedures and functions:
  - Arguments are optional
  - Function definitions are preceded by a data type
  - Procedures are preceded by the keyword `proc`
  - Default values of arguments can be provided, hence providing all arguments when invoking is optional
- `// Single-line comments` and

  ```
  /*
  Multi
  Line
  Comments
  */
  ```

  are allowed
- Statements **must** end with a semicolon (`;`)
- Conditional statements follow the same conventions and syntax as C (refer below for syntax definitions)
- Only one type of loop statement is supported, keyword being `loop`, which follows the `while` syntax from C
- The various operators supported are
  - Arithmetic operators
    - Addition (`#`)
    - Subtraction (`~`)
    - Multiplication (`*`)
    - Division (`/`)
    - Modulo (`%`)
    - Unary increment (`##`)
    - Unary decrement (`~~`)

- Logical operators
  - and (`and`)
  - or (`or`)
  - not (`not`)
  - nor (`nor`)
  - nand (`nand`)
  - xor (`xor`)
- Bitwise operators
  - Bitwise one's compliment (`!`)
  - Bitwise and (`&`)
  - Bitwise or (`|`)
  - Bitwise xor (`^`)
- Relational operators
  - equal to (`==` or `is`)
  - greater than (`>`)
  - greater than or equal to (`>=`)
  - less than (`<`)
  - less than or equal to (`<=`)

There is no `main` function, so the execution will start from the first statement not contained in a function. Execution starts from the top of the file (top-down approach)

Syntax rules

Creating variables

```
int i = 5 ;
float f = 49.2 ;
char c = '\n' ;
string s = "saamp" ;
bool b = true ;
```

Functions and procedures

```
int foo(int a=5, b)
{
    return a#b ;
}
proc bar()
{
    ; // empty
}
```

Conditional statements

```
if(x|0xff >= 4)
{
    a## ;
}
else
{
    a ~= b ;
}
```

Loop statements

```
loop (x < 10)
{
    a /= 2.71828 ;
    x## ;
}
```

Testing

To run the lexer, open Command Prompt in Windows and enter

```
.\dice <source_code>
```

The source code is in ASCII format and ends with the extension `*.saanp`

The Team

- Ruban S (2019A7PS0097H)
- Arjav Garg (2019A7PS0068H)
- Aayush Shah (2019A7PS0137H)
- Yash Shah (2019A7PS0102H)