AKSHITA MITTEL
CS13B1040

COMPILER PROGRAMS

ASSIGNMENT 0: CS3020

The trivial programs were:

- fact.cl
- fib.cl
- triangle.cl
- primen.cl
- lcm.cl

The non-trivial programs were:

- hanoi.cl
- tictactoe.cl
- hangman.cl

PART 1:

The general correspondence between the COOL programs and the MIPS code are:

- All the objects classes that are in built in COOL are declared globally at first, such as the Main_protObj, Int_protObj and so on.
- The memory containers are defined next.
- All the objects, classes and data names such as the file name are stored in string constants, with their attributes being declared. This block also contains all the user defined strings, such as the string that are sent to the console to be printed.
- All the predefined integers are dded to the int dispatch table.
- All the objects, classes and other instances such as main are then defined, again, by setting their attributes. These attributes include, the member word that are included in the object. The memory heap is defined in the same way.
- Next the stack is initialized with the COOL program constructs, after-which our program starts.
- The IO class is initialized first, since our Main class inherits from it.
- The Main class is initialized, after-which the Main.main method is executed:
- In Recursive programs like fact.cl (fib.cl and hanoi.cl too), a label “main.Factorial” is defined, to loop between each recursion of the function.
- Predefined subroutines are used such as “equality_test”, which is used as a MIPS representation of (n = 1), the equality operation in the COOL program.
- Each branch of the if conditions are defined in a separate block, with unique labels that assist the transition from one block to another.

- The default labels “labelX” are used to create the iterative while loop.

PART 2:

A simple toy program was implemented. That indicates whether a number is odd or even using a simple for loop. The sole purpose of this program was to indicate the different types of errors inserted, as marked by the comments in each code.

Toy1.cl → Integers, identifiers, and Special notation
Change made: declared an int identifier as “n#”
Error shown: Wherever the int was called, it would not be able to identify the identifier correctly.

```
"toy1.cl", line 6: syntax error at or near ERROR = #
"toy1.cl", line 9: syntax error at or near ERROR = #
"toy1.cl", line 11: syntax error at or near ERROR = #
Compilation halted due to lex and parse errors
```

Toy2.cl → Strings
Change made: created a non-escaped newline
Error shown: The string made on two lines was interpreted as 2 different stings.

```
"toy2.cl", line 14: syntax error at or near ERROR = Unterminated string constant
Compilation halted due to lex and parse errors
```

Toy3.cl → Comments
Change made: Added a comment which wasn't properly nested.
Error shown: This indicates that the error was affiliated to the comments structure.

```
"toy3.cl", line 24: syntax error at or near ERROR = EOF in comment
Compilation halted due to lex and parse errors
```

Toy4.cl → Keywords
Change made: used an upper case for a keyword other than “true” or “false”
Error shown: The compiler failed to recognize the keyword THEN; hence showed a parse error.

```
"toy4.cl", line 21: syntax error at or near '}'
Compilation halted due to lex and parse errors
```

Toy5.cl → White space
Change made: Added a whitespace in the keyword String.
Error shown: it treated the “ing” from the “String” as a completely different object and hence showed a parse error.

```
 "toy5.cl", line 13: syntax error at or near OBJECTID = ing
Compilation halted due to lex and parse errors
```