Assignment 1 (Symbol Table)
Department of CSE, MIST

1   Introduction

The purpose of this course is to construct a very simple compiler. In the first step to do so, we are going to implement a symbol-table. A *symbol-table* is a data structure maintained by compilers in order to store information about the occurrence of various entities such as identifiers, objects, function names etc. Information of different entities may include type, value, scope etc. At the starting phase of constructing a compiler, we will not go into many detail. We will simply construct a symbol-table based on hashing where collision is resolved by chaining. Fig 1 illustrates a sample symbol table.
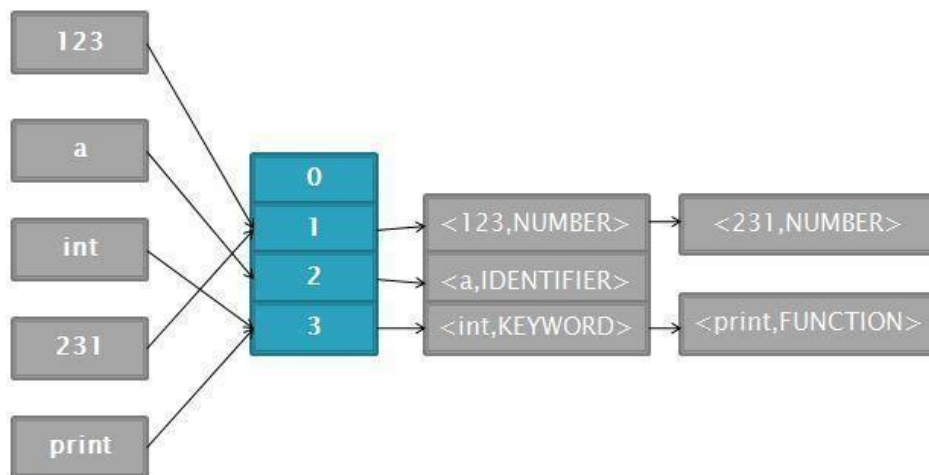

Figure 1: Symbol Table using hashing

2   Tasks

You have to implement following two classes.

- **SymbolInfo:** The definition of this class will grow gradually throughout the development of this project. For this assignment, we simply need two members, one for storing the symbol (e.g. "a") and another for storing the type of the symbol (e.g. "IDENTIFIER").

- **SymbolTable:** Since our symbol-table will be a hash-table based on chaining, we will have to start with an array of pointers where each pointer points to a list of nodes of type class SymbolInfo. Class **SymbolTable** will have such an array of pointers. For this assignment, the choice of the

size of this array, as well as of the hash function is left up to you. In addition to this array of pointers, class **SymbolTable** will have the following functions.

- **– Insert:** Insert into symbol table if already not inserted.

- **– Look up:** Search the hash table for particular symbol.

- **– Delete:** Delete an entry from the symbol table.

- **– Print:** Print the whole symbol table in the console.

Sample Input	Sample Output
I 123 NUMBER	Inserted at position 1,0
I 231 NUMBER	Inserted at position 1,1
I a IDENTIFIER	Inserted at position 2,0
I + OPERATOR	Inserted at position 4,0
P	0 -> 1 -> <123, NUMBER> <231, NUMBER> 2 -> <a, IDENTIFIER> 4 -> <+, OPERATOR>
L 123	Found at 1,0
D 123	Deleted from 1,0

Table 1: Sample Input Output

3 Input

Each line of the input will start with a code letter indicating the operation you want to perform. The letters will be among 'I', 'L', 'D' and 'P'. 'I' stands for insert which is followed by two space separated strings where the first one is symbol name and the second one is symbol type. As you might already guessed symbol name will be the key of the record to be stored in the symbol-table. 'L' means lookup which is followed by a string containing the symbol to be looked up in the table. 'D' stands for delete which is also followed by a string to be deleted. At last 'P' stands for print the symbol table.

4 Output

For insert command you have to print the array index of the hash table as well as the index of the chain where the symbol is placed by your program. If your program fail to insert then it should print reason of failure. For lookup or delete you have to print the position of look upped or deleted element just like the insert command if you can carry out the command, otherwise show appropriate error message. At last for print command, print the whole symbol table (you may skip the empty indexes). See Table 1 for more clarification. Check the sample io given in moodle and try to follow it.

5 Important Notes

Please try to follow the instructions listed below while implementing your assignment:

- Implement using C++ programming language
- Avoid hard coding
- Use dynamic memory allocation
- Take input from file. You may output both in console and file.

6 Rules

- Any type of plagiarism is strongly forbidden and may result in severe punishments.
- Prepare for an online evaluation.

Hash Technique for Task

Technique-1: For each of the symbol, take the ASCII value of the middle character of symbol and multiply it with the last two digits of your own roll number. At last MOD it with the MOD value and get the indexing value in the symbol table.

Symbol	ASCII Value of the Middle Character	Mul with Roll No	MOD	Indexing Value
for	111	111 * 27 = 2997	2997 % 101 = 68	68

Technique-2: For each of the symbol, take the ASCII value of the first character of symbol and multiply it with the today's date plus last two digit of your own roll number. At last MOD it with the MOD value and get the indexing value in the symbol table.

Symbol	ASCII Value of the Starting Character	Mul with today's date	MOD	Indexing Value
for	102	102 * (25+25) = 2550	2997 % 100 = 25	25