**Name: Sparsh Karna**

**Reg Number: 23BDS1172**

**Lab Excercise: 3**

**Aim:**

The aim of this program is to process variable declarations from a file, generate a symbol table containing valid entries, and create an error table containing invalid or duplicate entries.

**Procedure:**

1. The program reads each line from a text file (lab3\_23BDS1172.txt) containing variable declarations in the format:

type identifier value

1. The process involves:
   1. Ignoring empty lines.
   2. Splitting each line into three parts - data type, identifier name, and value.
   3. Checking if the declaration format is correct.
   4. Checking for duplicate identifiers.
   5. Storing valid entries in the symbol table along with their index, type, and value.
   6. Storing invalid entries in the error table with a reason for rejection.
2. The symbol table and error table are written to separate text files:
   1. symbol\_table.txt
   2. error\_table.txt

**Algorithm:**

1. Open lab3.txt for reading.
2. Open symbol\_table.txt and error\_table.txt for writing results.
3. Initialize a set to keep track of already declared identifiers.
4. Initialize a vector to store symbol table entries.
5. Initialize a vector to store error table entries.
6. For each non-empty line in the input file:
   1. Parse the data type, identifier, and value.
   2. If parsing fails, record the identifier in the error table with the reason "Malformed declaration"
   3. If the identifier is already in the set, record it in the error table with the reason "Identifier already declared".
   4. Otherwise, insert it into the set and add a new entry to the symbol table.
7. Write the symbol table to symbol\_table.txt.
8. Write the error table to error\_table.txt.
9. Display a confirmation message on the console.

**Program:**

1. **Code:**

**#include <iostream>**

**#include <fstream>**

**#include <sstream>**

**#include <vector>**

**#include <unordered\_set>**

**#include <unordered\_map>**

**using namespace std;**

**struct SymbolEntry**

**{**

**int id;**

**string name;**

**string datatype;**

**string assignedValue;**

**};**

**struct ErrorEntry**

**{**

**string name;**

**string message;**

**};**

**int main()**

**{**

**ifstream inputFile("lab3\_23BDS1172.txt");**

**ofstream symFile("symbol\_table.txt");**

**ofstream errFile("error\_table.txt");**

**if (!inputFile)**

**{**

**cerr << "Error: lab3.txt not found!\n";**

**return 1;**

**}**

**unordered\_set<string> seenIdentifiers;**

**vector<SymbolEntry> symbols;**

**vector<ErrorEntry> errors;**

**string currentLine;**

**int symbolIndex = 1;**

**while (getline(inputFile, currentLine))**

**{**

**if (currentLine.find\_first\_not\_of(" \t\r\n") == string::npos)**

**continue;**

**istringstream parser(currentLine);**

**string type, name, value;**

**if (!(parser >> type >> name >> value))**

**{**

**errors.push\_back({name, "Malformed declaration"});**

**continue;**

**}**

**if (seenIdentifiers.count(name))**

**{**

**errors.push\_back({name, "Identifier already declared"});**

**continue;**

**}**

**seenIdentifiers.insert(name);**

**symbols.push\_back({symbolIndex++, name, type, value});**

**}**

**symFile << "Index\tIdentifier\tType\tValue\n";**

**for (auto &entry : symbols)**

**{**

**symFile << entry.id << "\t" << entry.name << "\t"**

**<< entry.datatype << "\t" << entry.assignedValue << "\n";**

**}**

**errFile << "Identifier\tReason\n";**

**for (auto &entry : errors)**

**{**

**errFile << entry.name << "\t" << entry.message << "\n";**

**}**

**cout << "Symbol table and error table have been created successfully.\n";**

**return 0;**

**}**

1. **Lab3\_23BDS1172.txt:**

int a 10

float b 3.14

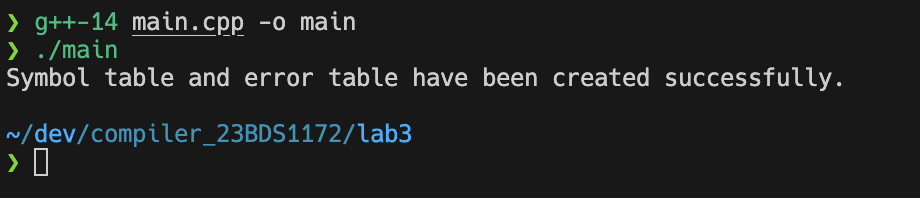
string name Ram

int a 20

char c x

**Output:**

**Terminal Output:**



**symbol\_table.txt:**

**Index** **Identifier** **Type** **Value**

**1** **a** **int** **10**

**2** **b** **float** **3.14**

**3** **name** **string** **Ram**

**4** **c** **char** **x**

**error\_table.txt:**

**Identifier** **Reason**

**a** **Identifier already declared**

**Result:**

The program successfully generated a symbol table containing valid entries and an error table for duplicate or incorrectly formatted declarations, ensuring accurate tracking of variables for further compilation stages.