**Compiler Project Report**

**On Mini C++ Compiler**

Submitted By :-

Vikhyat Tandon IIT2011203

Ankit Jat IIT2011205

Atul Sakhala IIT2011206

Atul Vaibhav IIT2011208

Vikash Sandhu IIT2011209

Shubham Mishra IIT2011211

1. Problem statement

* Construct a mini C++ type compiler.
* It should be able to strictly identify only C++ code.
* It should report an error And any C code which is acceptable in C++.

2. Introduction

The C and C++ programming languages are closely related. C++ grew out of C, as it was designed to be source-and-link compatible with C. C++ was based on C and retains a great deal of the functionality. The C++ language provides mechanisms for mixing code that is compiled by compatible C and C++ compilers in the same program. As a matter of fact, C++ can run most of C code while C cannot run most C++ code.

The **purpose of compatibility** with C is so that C++ programs can have **convenient access** to the billions (trillions?) of lines of existing C code in the world.

Although, C and C++ code are almost compatible but there are still many incompatibilities or conflicts between them. The conflicts can be of two types:

1. **Incompatible C feature** - valid as C code but not as C++ code.
2. **Incompatible C++ feature** - valid as C++ code but not as C code.

In this project we focus on a different domain. **Compatible** C/C++ features i.e. features of C code that are valid in C++.

We aim at detecting such snippets of code in our input program and will give an error if a C code is detected, whilst if no C code could be detected then we will compile it for minor errors, i.e. **a mini compiler strictly for C++.**

An example of C code valid in C++ code:

#include <iostream>

#include <cstdio>

#include <cmath>

using namespace std;

int main()

{

int x;

// C++ style IO

cout << “Hello World, Enter a number: ” << endl;

cin >> x;

/\* C Style IO \*/

printf(“Hello World, Enter a number: ”);

scanf(“%d”, &x);

/\* A fuction call invalid in C++ \*

\* if not including C header files. \*/

x = sqrt(4);

return 0;

}

In addition to detect C code inside C++ code, our Mini C++ compiler will also be able to report following errors to the user:

* Invalid variable name.
* Invalid basic arithmetic expression.
* Syntax error in While loop.
* Syntax errors in For loop.
* Syntax errors in If-Then-Else.

3. Tools used

We have used following two tools to implement our project:

1. YACC - produces a parser
2. LEX - generates lexical analyzers
3. GCC – gnu C Compiler
4. G++ - gnu C++ Compiler

4. Methodology or Algorithm

The project is implemented in the following steps: -

1. Read the given Input.
2. Tokenize the input using Lex rules.
3. Parse using Yacc rules.
4. Run the Algorithm described below.

Algorithm:-

Goal: detect a c code that is generally successfully compiled by a C++ compiler and accept small C++ codes.

Steps:

1. Detect for header files (generally all c codes have **.h** header files)
2. Detect C language functions and keywords that are compatible with C++ compilers.
3. If(c code detected )

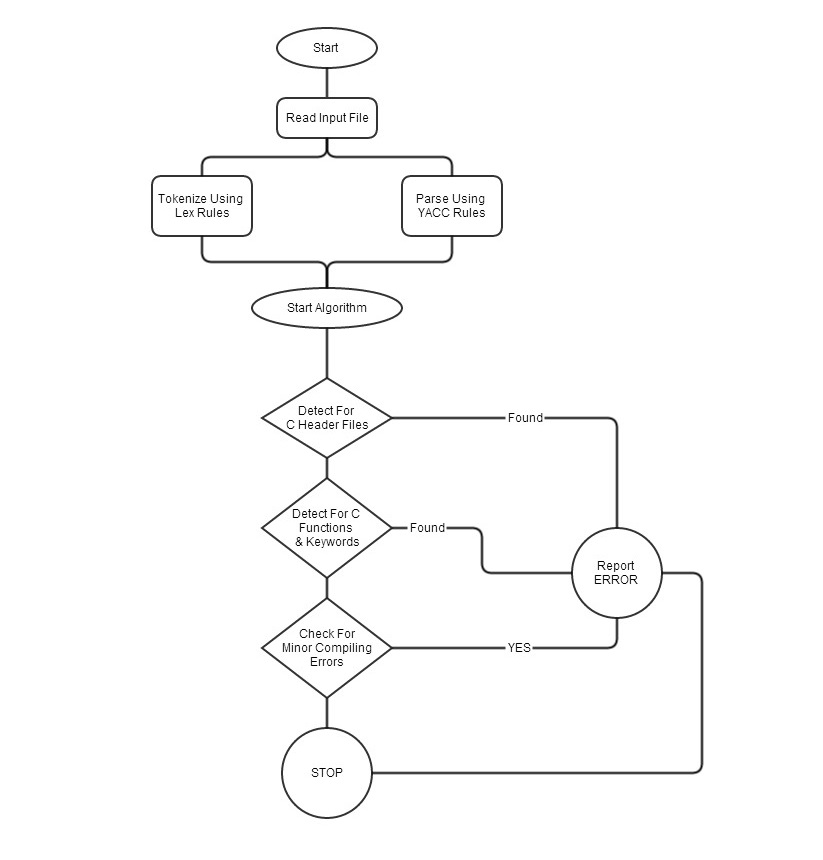
Then return ERROR and STOP.

Else goto step 4.

4. Check for error like

* + - * + Invalid variable name.
        + Invalid basic arithmetic expression.
        + Syntax error in While loop.
        + Syntax errors in For loop.
        + Syntax errors in If-Then-Else.

5. Flowchart



6. Work done so Far

1. Did the required literature survey.

2. Design our Algorithm

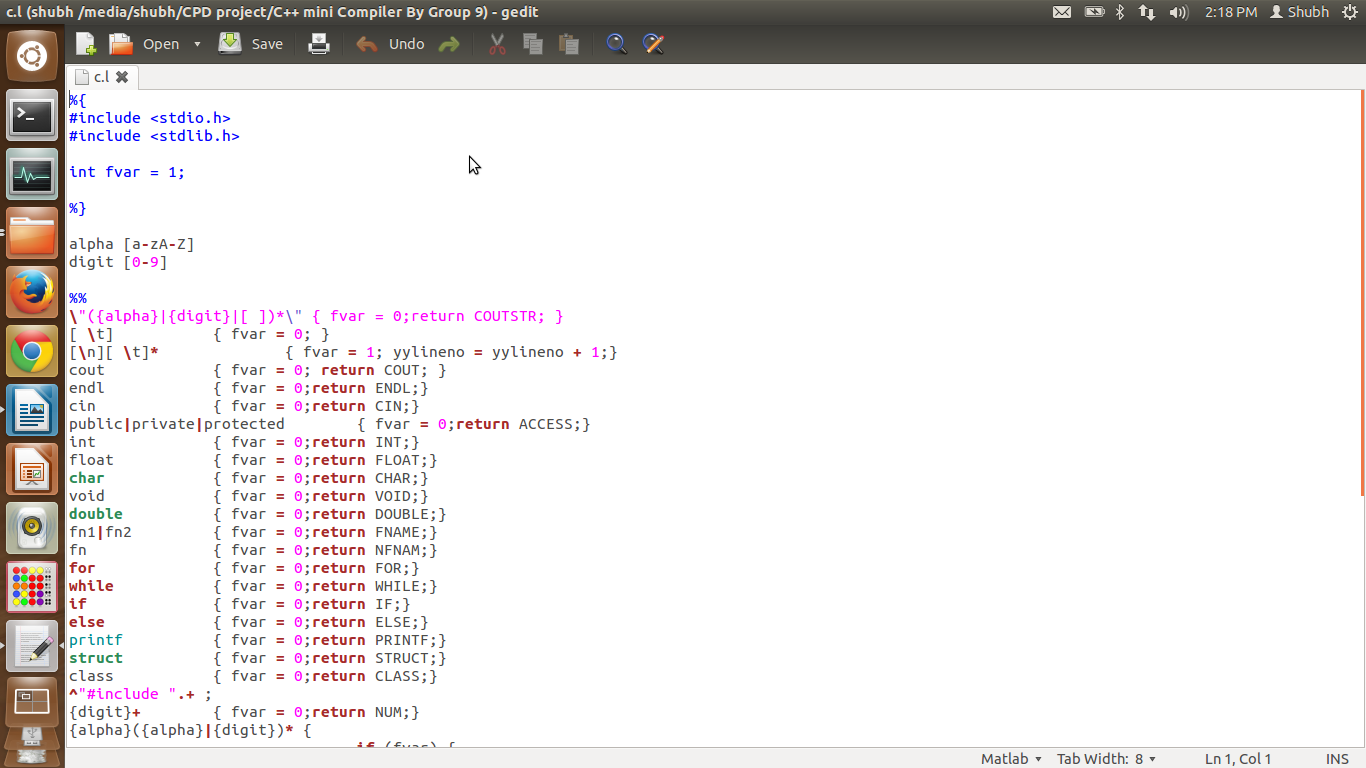
3. Flow chart prepared.

4. Project report 1 created.

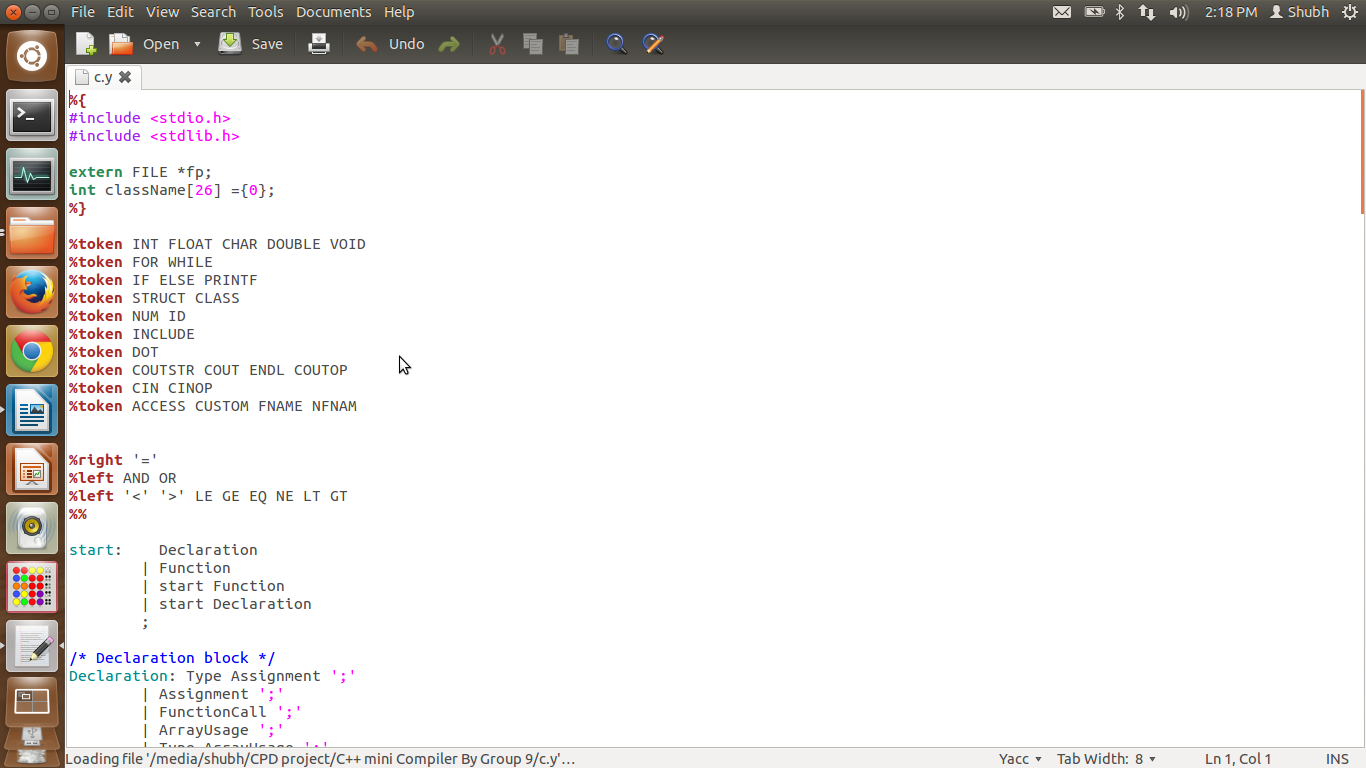
5. Implemented the mini compiler for C++.

6. Completed the project with results verified with test files.

Lex File Snap

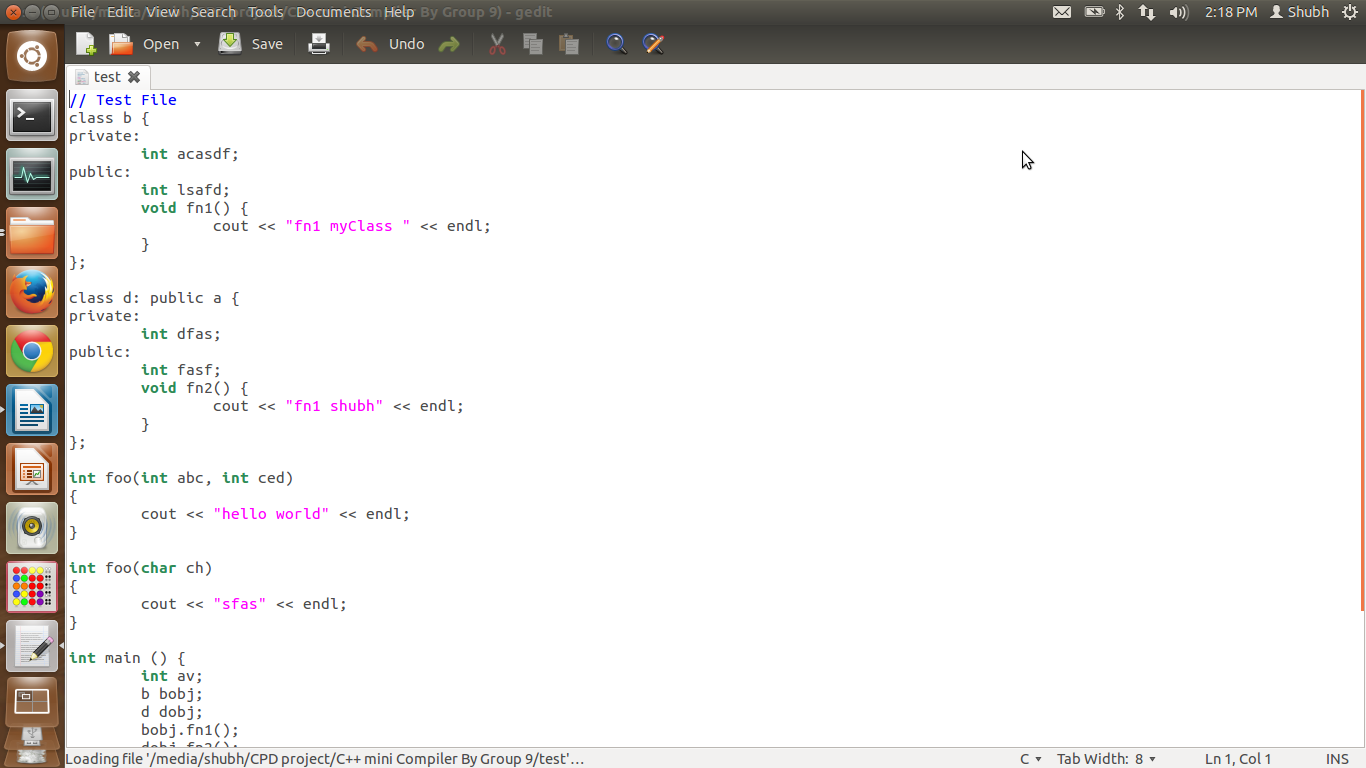


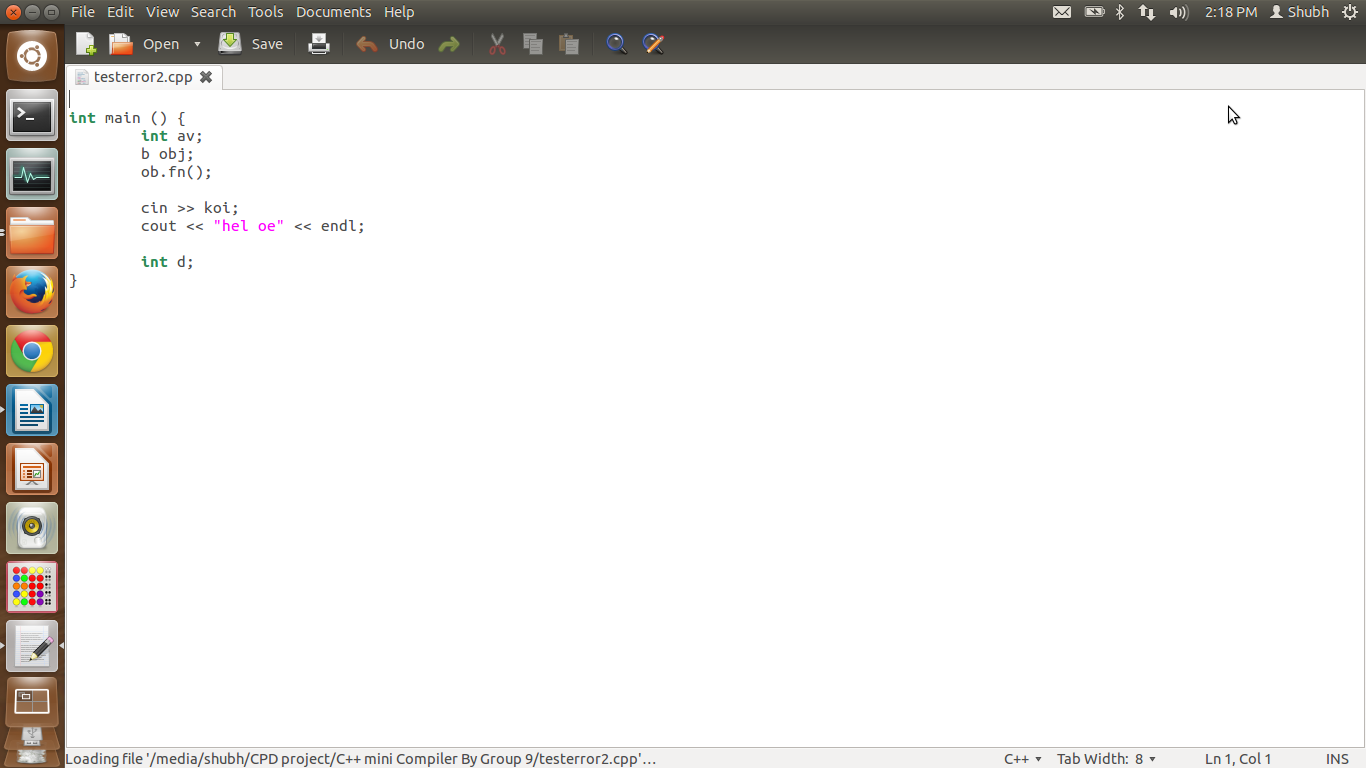
Yacc File Snap



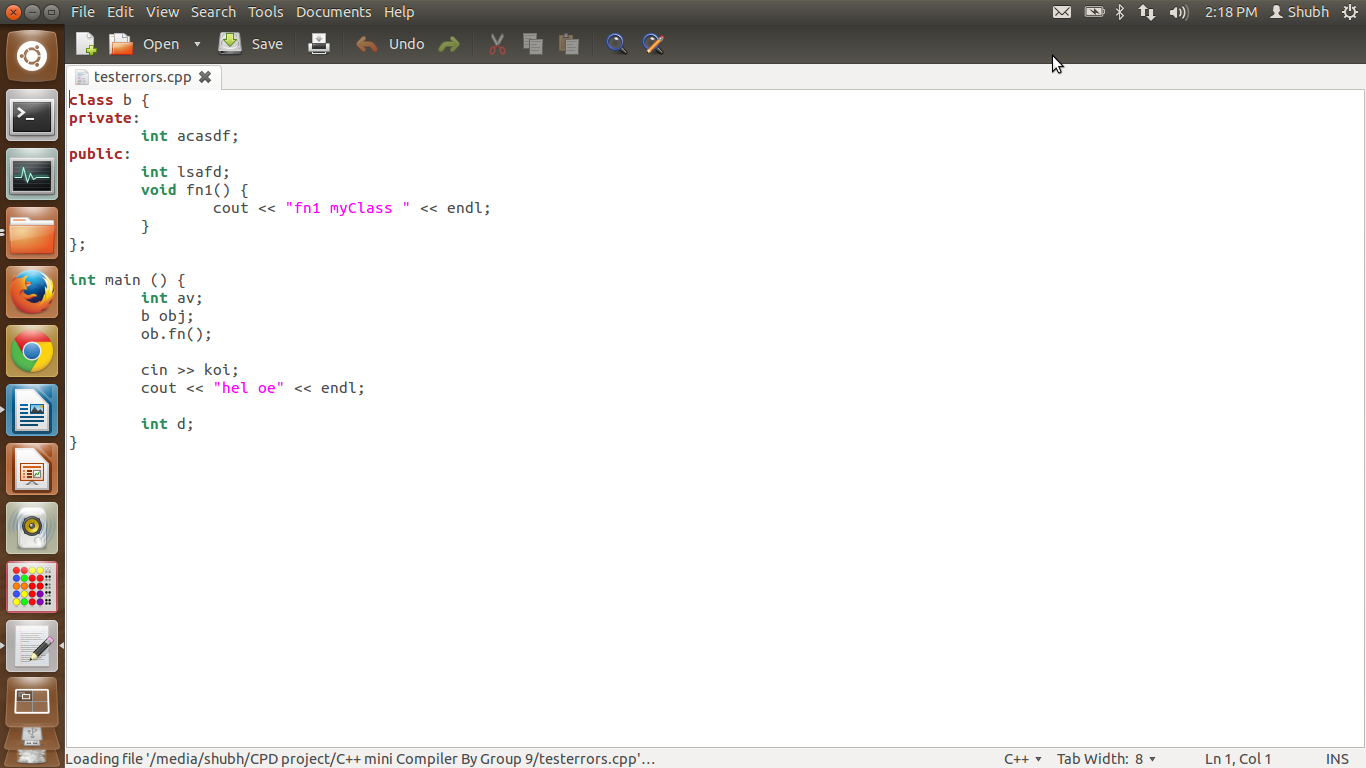
Test File For Polymorphism:-

Test File for error





Test file for error



\_\_\_\_\_\_\_\_\_THANKS\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_