```
#include<iostream>
#include<regex>
#include<vector>
#include<fstream>
using namespace std;
string stripL(string );
string stripR(string );
string strip(string );
string get_subString(string ,char ,char );

void error_msg_show(int line,string message){
   cout<<"Error found in line : "<<line<<endl;
   cout<<"Error message : "<<message<<endl<<endl;
}

int main(){
   regex regex_header_start("(#\\s*include\\s*)(.*)");
      regex regex_header_line_valid("(#\\s*include\\s*<)(.*)>");
   regex regex_main("(.*) main(.*)");
      regex regex_main_parm("(.*) main()(.*)");
         regex regex_main_return_type_void("\\s*void(.*)");
         regex regex_main_return_type_int("\\s*int(.*)");
   regex regex_return_statment("(.*)return\\s(.*)");
      regex regex_return_int("(.*)return\\s*(0|-?([1-9][0-9]*)\\s*);");
   regex regex_digit("[0-9]");
   regex regex_digits("([1-9][0-9]*)");
   regex regex_scop_start("(.*)\\{(.*)");
   regex regex_scop_end("(.*)\\}(.*)");

   string fileName="001source_code.c";
   ifstream fs;
   fs.open(fileName);
   if (!fs) {
      cout << "Unable to open file";
      exit(1);
   }
   bool multilineCommentOut{false};
   bool return_need{false};
   bool scop_start{false};
   int lineNumber=1;
```

```
string line;
smatch match;
while(getline(fs,line)){
   start:
   line = strip(line);
   cout<<"After strip line is : \""<<line<<"\""<<endl;

   ///Skiping line checking
   if(multilineCommentOut){
      cout<<"Line commented"<<endl;
      if(regex_search(line,match,regex("\\*/"))){
         cout<<"Finished comment"<<endl;
         line=match.suffix();
         multilineCommentOut=false;
         goto start;
      }continue;
   }else if(line==""||regex_match(line,regex("^//.*"))){
      cout<<"empty line or comment line"<<endl;
      continue;
   }
   else if(regex_search(line,match,regex("//"))){
      cout<<"sub comment line"<<endl;
      line = strip(match.prefix());
      cout<<"Remaining line is : \""<<line<<"\""<<endl;
   }else if(regex_search(line,match,regex("/\\*"))){
      cout<<"MultiLine comment out found"<<endl;
      line = match.prefix();
      multilineCommentOut=true;
      string commentString=match.suffix();
      if(regex_search(commentString,match,regex("\\*/"))){
         cout<<"Finish in same line"<<endl;
         line+=match.suffix();
         multilineCommentOut=false;
         goto start;
      }
      if(line==""){
         continue;
      }
   }
```

```
    cout<<"Processing line is : \""<<line<<"\""<<endl;
    bool isvalidLine=true;
    if(regex_match(line,regex_scop_start)){
       scop_start=true;
    }
    if(regex_match(line,regex_header_start)){
       cout<<"Header found"<<endl;
       if(regex_match(line,regex_header_line_valid)){
          cout<<"A Valid header found"<<endl;
          string header = get_subString(line,'<','>');
          header=strip(header);
          cout<<"Header is :\""<<header<<"\""<<endl;
          if(header!="stdio.h"){
             error_msg_show(lineNumber,header+" not defined header.");
             return 0;
          }
       }else{
          error_msg_show(lineNumber,"Invalid header diclaration.");
          return 0;
       }
    }else if(regex_match(line,regex_main)){
       cout<<"Main found."<<endl;
       if(regex_match(line,regex_main_parm)){
          cout<<"Main function found."<<endl;
          if(regex_match(line,regex_main_return_type_int)){
             cout<<"Main should be return an integer."<<endl;
             return_need = true;
          }else if(regex_match(line,regex_main_return_type_void)){
             cout<<"Main return not need."<<endl;
          }else{
             error_msg_show(lineNumber,"Invalid return type of Main
function.");
             return 0;
          }
       }else{
          error_msg_show(lineNumber,"Invalid main function declaration.");
          return 0;
       }
    }
```

```
        else if(regex_match(line,regex_return_statment)){
            cout<<"Return statement found"<<endl;
            if(!scop_start){
                error_msg_show(lineNumber,"Not in a scope.");
                return 0;
            }if(regex_match(line,regex_return_int)){
                cout<<"Return type an integer found."<<endl;
                return_need = false;
            }else{
                error_msg_show(lineNumber,"Return type not an integer.");
                return 0;
            }
        }else{
            isvalidLine = false;
        }
        if(regex_match(line,regex_scop_end)){
            scop_start=false;
            isvalidLine = true;
        }
        if(!isvalidLine){
            error_msg_show(lineNumber,"Invalid statement found.");
            return 0;
        }
        lineNumber++;
    }
    fs.close();
    if(return_need){
        error_msg_show(lineNumber,"Return not found in the scope.");
        return 0;
    }
    if(scop_start){
        error_msg_show(lineNumber,"Scope not finished.");
        return 0;
    }
    cout<<"**********Compilation success.**********"<<endl;
    return 0;
}
```

```
string stripL(string input_str){
   int starting_pointer=0;
   while(input_str[starting_pointer]==' '){
      starting_pointer++;
   }
   return input_str.substr(starting_pointer);
}
string stripR(string input_str){
   int ending_pointer=input_str.size()-1;
   while(input_str[ending_pointer]==' '){
      ending_pointer--;
   }
   return input_str.substr(0,ending_pointer+1);
}
string strip(string input_str){
   string part = stripL(input_str);
   part = stripR(part);
   return part;
}
string get_subString(string str,char start_dilam,char end_dilam){
   stringstream ss(str);
   string part;
   getline(ss,part,end_dilam);
   ss = stringstream(part);
   getline(ss,part,start_dilam);
   getline(ss,part,start_dilam);
   return part;
}
```