```cpp
#include <iostream>
#include <fstream>
#include <sstream>
#include <vector>

using namespace std;

class file
{

string dir;

public:
file(string dir) : dir(dir) {}

bool hasComment(string line)
{
for (int i = 0; i < line.length(); i++)
if (line[i] == '/' && line[i + 1] == '/')
return true;
return false;
}

void commentRemove(string &line)
{
bool flag = hasComment(line);
while (flag)
{
if (line[line.length() - 1] != '/')
line.pop_back();
else
{
line.pop_back();
line.pop_back();
break;
}
}
}

bool isNonWhiteSpace(string line)
{
for (char ch : line)
{
if (!isspace(ch))
return true; // Found a non-whitespace character
}
return false; // Line contains only whitespace characters
}

vector<string> extrator()
{
```

```
vector<string> line;
ifstream fin(dir);
if (fin.is_open())
{
while (!fin.eof())
{
string temp;
while (!isNonWhiteSpace(temp) && !fin.eof())
getline(fin, temp);

commentRemove(temp);

istringstream iss(temp);
string word;
while (iss >> word)
line.push_back(word);
}

fin.close();
}

else
cout << "file is failed to open" << endl;

return line;
}

void libraryCheck(vector<string> &line)
{
string temp;
if (line[0] == "#")
{
temp = line[0];
line.erase(line.begin());
if (line[0] == "include")
{
temp += line[0];
line.erase(line.begin());
}
}
else if (line[0] == "#include")
{
temp = line[0];
line.erase(line.begin());
}
if (line[0] == "<iostream>")
{
temp += " " + line[0];
line.erase(line.begin());
}
```

```
if (temp == "#include <iostream>")
cout << temp << " ---> Dependency" << endl;
else
cout << "No library found" << endl;
}

void namespaceCheck(vector<string> &line)
{
string temp;
if (line[0] == "using")
{
temp = line[0];
line.erase(line.begin());
}
if (line[0] == "namespace")
{
temp += " " + line[0];
line.erase(line.begin());
}
if (line[0] == "std")
{
temp += " " + line[0];
line.erase(line.begin());
if (line[0] == ";")
{
temp += line[0];
line.erase(line.begin());
}
}
else if (line[0] == "std;")
{
temp += " " + line[0];
line.erase(line.begin());
}

if (temp == "using namespace std;")
cout << temp << " ---> namespace" << endl;
else
cout << "No namespace found" << endl;
}

void functionCheck(vector<string> &line)
{
string temp;
if (line[0] == "int")
{
temp += line[0];
line.erase(line.begin());
}
if (line[0] == "main")
{
```

```
temp += " " + line[0];
line.erase(line.begin());

if (line[0] == "(")
{
temp += line[0];
line.erase(line.begin());

if (line[0] == ")")
{
temp += line[0];
line.erase(line.begin());
if (line[0] == "{")
{
temp += line[0];
line.erase(line.begin());
}
}

else if (line[0] == "){")
{
temp += line[0];
line.erase(line.begin());
}
}
else if (line[0] == "()")
{
temp += line[0];
line.erase(line.begin());
if (line[0] == "{")
{
temp += line[0];
line.erase(line.begin());
}
}
else if (line[0] == "(){")
{
temp += line[0];
line.erase(line.begin());
}
}

else if (line[0] == "main(")
{
temp += " " + line[0];
line.erase(line.begin());

if (line[0] == ")")
{
temp += line[0];
line.erase(line.begin());
```

```
if (line[0] == "{")
{
temp += line[0];
line.erase(line.begin());
}
}
else if (line[0] == "){")
{
temp += line[0];
line.erase(line.begin());
}
}
else if (line[0] == "main()")
{
temp += " " + line[0];
line.erase(line.begin());
if (line[0] == "{")
{
temp += line[0];
line.erase(line.begin());
}
}

if (temp == "int main(){")
cout << temp << " ---> Main function declaration" << endl;
else
cout << "No main function found" << endl;
}

bool isDataType(string word)
{
ifstream fin("DataType.txt");
if (fin.is_open())
{
string line;
while (!fin.eof())
{
fin >> line;
if (line == word)
{
return true;
break;
}
}
}
else
cout << "file is failed to open" << endl;
return false;
}

bool validity1(string statement)
```

```
{
int count = 0;
istringstream iss(statement);
string word;
while (iss >> word)
if (isDataType(word))
count++;

if (count > 1)
return true;

return false;
}
void operatorCheck(vector<string> value)
{
if (value.size() == 1)
{
for (string v : value)
cout << v << " ";
cout << endl;
}
else
{
if (value[1] == "+")
cout << "Summation of " << value[0] << " and " << value[2] << endl;
else if (value[1] == "-")
cout << "Substraction of " << value[0] << " and " << value[2] << endl;
else if (value[1] == "*")
cout << "Multiplication of " << value[0] << " and " << value[2] << endl;
else if (value[1] == "/")
cout << "Division of " << value[0] << " and " << value[2] << endl;
else if (value[1] == "%")
cout << "Mod of " << value[0] << " and " << value[2] << endl;
}
}

void validity2(string statement)
{
vector<string> temp;
vector<string> value;
istringstream iss(statement);
string word;
int c = 0;

while (iss >> word)
{
temp.push_back(word);
}

for (int i = 1; i < temp.size(); i++)
{
```

```
if (!isDataType(temp[i]) && temp[i] != "," && temp[i] != "=")
{
if (temp[i - 1] != "=" && isDataType(temp[i - 1]))
{
if (isValidIdentifier(temp[i]) && keywordCheck(temp[i]))
cout << temp[i] << " variable is " << temp[i - 1] << " type";
if (temp[i + 1] == "=")
{
cout << " which value is ";
i += 2;
while (temp[i] != "," && i < temp.size())
{
value.push_back(temp[i]);
i++;
}
operatorCheck(value);
}
}
else if (!isDataType(temp[i - 1]) && isValidIdentifier(temp[i]))
{
if (isValidIdentifier(temp[i]) && keywordCheck(temp[i]))
cout << " data type is not correct ";
if (temp[i + 1] == "=")
{
cout << " which variable is " << temp[i] << " and value is ";
i += 2;
while (temp[i] != "," && i < temp.size())
{
value.push_back(temp[i]);
i++;
}
operatorCheck(value);
}
}
else if (!isDataType(temp[i - 1]) && !isValidIdentifier(temp[i]))
{
cout << " and data type is not correct ";
break;
}
}
}
cout << endl;
}

bool keywordCheck(string text)
{
text += ",";
ifstream fin("keywords.txt");
if (fin.is_open())
{
```

```
string ch;
int i = 0;
while (!fin.eof())
{
fin >> ch;
if (ch == text)
{
cout << "Invalid variable name" << endl;
return false;
}
}
return true;
}
else
{
cout << "file is failed to open" << endl;
return false;
}
}

void filtering(string &statement)
{
for (int i = 0; i < statement.length(); i++)
{
if (statement[i] == '=' || statement[i] == ',')
{
if (statement[i - 1] != ' ')
{
statement.insert(statement.begin() + i, ' ');
i++;
}
if (statement[i + 1] != ' ')
{
statement.insert(statement.begin() + i + 1, ' ');
}
}
}
}

bool isValidIdentifier(string literal)
{
if (literal[0] >= 65 && literal[0] <= 90 || literal[0] >= 97 && literal[0] <= 122 || literal[0] == 95)
{
for (char value : literal)
{
if (!(value >= 65 && value <= 90 || value >= 97 && value <= 122 || value == 95 || value >= 48 && value <= 57))
{
if (value == ' ')
cout << "Invalid Character 'Space' in variable name and ";
else
cout << "Invalid Character in variable name and " << value << " ";
return false;
}
}
return true;
}
else
{
cout << "Invalid Character in variable name " << literal[0] << " ";
return false;
}
}

void variableCheck(vector<string> &line)
{
string statement = statementExtract(line);
filtering(statement);

if (validity1(statement))
{
bool flag = false;
string temp;
istringstream iss(statement);
string word;
iss >> word;
cout << word << " ";
while (iss >> word)
{
if (isDataType(word))
{
cout << " ---> Syntax Error : Semicolon is missing" << endl;
flag = true;
}
if (!flag)
cout << word << " ";
else
temp += word + " ";
}
cout << temp << " ---> ";
validity2(temp);
}
else
{
cout << statement << " ---> ";
validity2(statement);
}
}

bool check(vector<string> &line)
{
bool flag = false;
if (isDataType(line[0]))
```

```cpp
{
variableCheck(line);
flag = true;
}
else if (line[0] == "cout")
{
int c = 0, x = 0;
vector<string> st;
string statement, left, next;
string temp = statementExtract(line);

for (int i = 0; i < temp.size(); i++)
{
statement += temp[i];
if (temp[i] == '"' && c == 1)
{

for (int j = i + 1; j < temp.size(); j++)
{
left += temp[j];
if (temp[j] == 'l')
{
for (int k = j + 1; k < temp.size(); k++)
{
next += temp[k];
x = 1;
}
if (x == 1)
{
istringstream iss(next);
string word;
iss.ignore();
while (iss >> word)
{
st.push_back(word);
}
break;
}
}
}
break;
}
if (temp[i] == '"')
c = 1;
}

filtering(statement);
if (x == 0)
cout << statement << " ---> output is " << output(statement) << endl;

else if (x == 1)
```

```
{
cout << statement << " ---> output is " << output(statement);
cout << " and syntax error: semicolon missing" << endl;
check(st);
}
flag = true;
}
else if (line[0] == "cin")
{
string statement = statementExtract(line);
filtering(statement);
cout << statement << " ---> user input is " << input(statement) << endl;
flag = true;
flag = true;
}
else if (line[0] == "if")
{
string statement = statementExtract(line);
filtering(statement);
cout << statement << " } ---> it is if condition " << endl;
flag = true;
}
else if (!isDataType(line[0]))
{
variableCheck(line);
flag = true;
}

return flag;
}

string statementExtract(vector<string> &line) //showing error
{
string temp;
bool flag = false;
while (true)
{
for (int j = 0; line[0].size() > 0; j++)
{
if (line[0][j] == ';')
{
line[0].erase(line[0].begin() + j);
flag = true;
break;
}
else
{
if (line[0][j] != '}')
temp.push_back(line[0][j]);
line[0].erase(line[0].begin() + j);
j--;
```

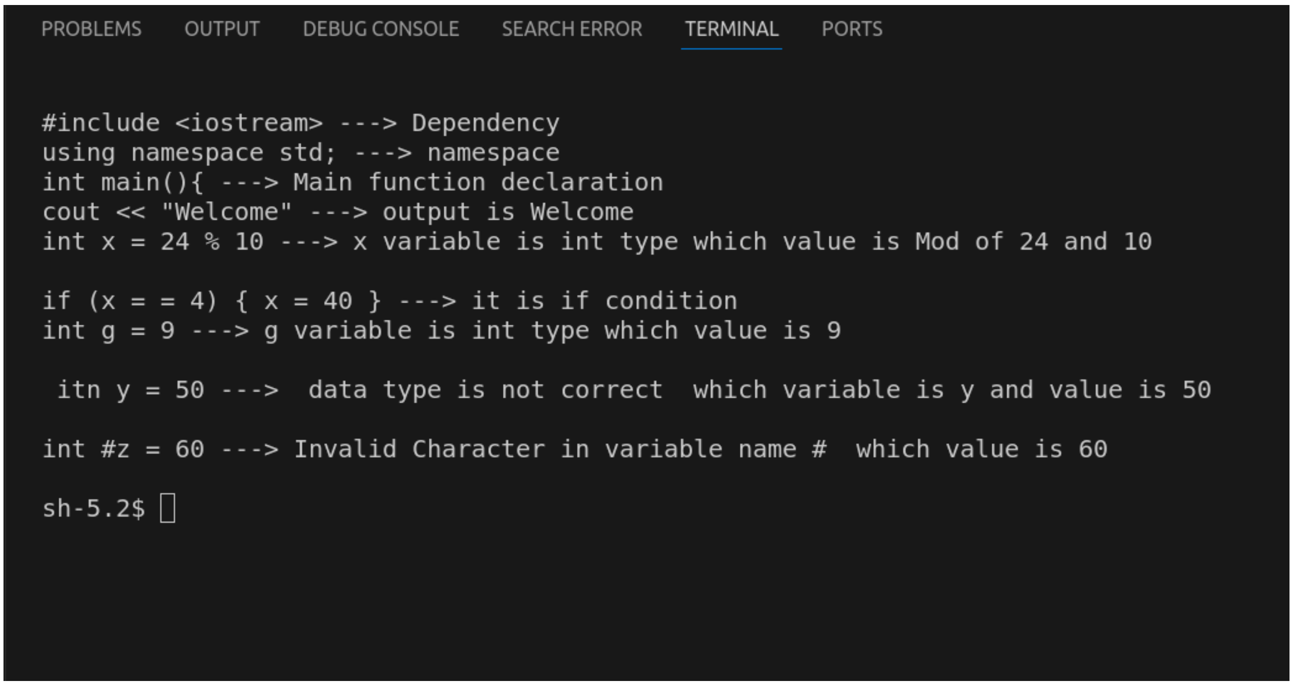
```
}
}
if (flag)
{
if (line[0] == "")
line.erase(line.begin());
break;
}
temp.push_back(' ');
if (line[0].size() == 0)
{
line.erase(line.begin());
if (line[0].size() == 0)
break;
}
}
return temp;
}

string output(string line)
{
string temp;
for (int i = 0; i < line.size(); i++)
{
if (line[i] == '"')
{
for (int j = i + 1; j < line.size(); j++)
{
if (line[j] == '"')
break;
temp += line[j];
}
break;
}
}
return temp;
}
string input(string line)
{
string temp;
for (int i = 0; i < line.size(); i++)
{
if (line[i] == '>')
{
for (int j = i + 2; j < line.size(); j++)
{
if (line[j] == ';' || isDataType(line))
break;
else
temp += line[j];
}
```

```
break;
}
}
return temp;
}

void preprocessor()
{
vector<string> line = extrator();
libraryCheck(line);
namespaceCheck(line);
functionCheck(line);
for (size_t i = 0; i < line.size(); ++i)
{
if (!check(line))
{
cout << "Unable to process line: " << line[i] << endl;
}
}
}
};

int main() // main fun declaration
{
string text;
file *f = new file("input.txt");
f->preprocessor();
delete f;
}
```



```
#include <iostream> ---> Dependency
using namespace std; ---> namespace
int main(){ ---> Main function declaration
cout << "Welcome" ---> output is Welcome
int x = 24 % 10 ---> x variable is int type which value is Mod of 24 and 10

if (x = = 4) { x = 40 } ---> it is if condition
int g = 9 ---> g variable is int type which value is 9

 itn y = 50 --->  data type is not correct  which variable is y and value is 50

int #z = 60 ---> Invalid Character in variable name #  which value is 60

sh-5.2$ 
```