Group - 05

```
#include<bits/stdc++.h>
using namespace std;
#define ll long long int
void insideClass(string);
void detectClass(void);
int blankline= 0;
bool commentOn = false;
string str = "";
int i = 0;
int current_line=0;
string SPACE = " \n\r\t\f\v";
set<string> className;
set<int> obj;
set<int> classDec;
set<int> constDec;
set<int> inherClass;
set<int> opOverload;
string keyword_class = "class";
string keyword_operator = "operator";
string keyword_return = "return";

void nextLine();

string removespace(string s)
{
    int start = s.find_first_not_of(SPACE);

    if (start==-1)
        return s.substr(0 , 0);

    s = s.substr(start);
    int end = s.find_last_not_of(SPACE);
    s = s.substr(0 , end + 1);
    return s;
}

void define_variables(string ans)
{
    ans = removespace(ans);
    string word;
    int j=0;
    if (ans[0]=='#')
    {
        j++;
        while(ans[j]==' '){
            j++;
        }
        while(j!=ans.size() && isalnum(ans[j])){
            word += ans[j];
            j++;
        }
        nextLine();
        if(word =="define"){
            word = "";
            while(ans[j]==' '){
                j++;
            }
            while(j!=ans.size() && isalnum(ans[j])){
                word += ans[j];
                j++;
            }
            nextLine();
            string word1;
            while(ans[j]==' '){
                j++;
            }
            while(j!=ans.size() && isalnum(ans[j])){
                word1 += ans[j];
                j++;
            }
            nextLine();
            if (word1 == "class")
                keyword_class = word;
            else if(word1 == "operator")
                keyword_operator = word;
            else if(word1 == "return")
                keyword_return = word;
        }
    }
    return;
}

string trim(string s)
{
      s = removespace(s);
    int j = 0;
    int slash;
```

```
string ans;
while(j<s.size()){
    if(commentOn){
        while(j<s.size() && s[j]!='*'){
            j++;
        }
        if(j==s.size() || j+1==s.size()){
            define_variables(ans);
            return ans;
        }
        if(s[j+1]=='/'){
            commentOn = false;
            j+=2;
        }
    }
    while(j<s.size() && (s[j]!='/' && s[j]!='"')){
        ans+=s[j];
        j++;
    }
    if(j==s.size()){
        define_variables(ans);
        return ans;
    }else if(j+1==s.size()){
        ans+=s[j];
        define_variables(ans);
        return ans;
    }
    if(s[j]=='/'){
        if(s[j+1]=='/'){
            define_variables(ans);
            return ans;
        }else if(s[j+1]=='*'){
            commentOn = true;
            j+=2;
        }
        else j++;
    }else{
        j++;
        while(s[j]!='"'){
            if(s[j]=='\\'){
                j+=1;
            }
            j++;
        }
        j++;
    }
    }
    ans = removespace(ans);
    define_variables(ans);
    return ans;
}

void nextLine(){

    while(str.size()==i){
        if(!getline(cin, str)){
            cout<<"Object Declaration - "<<obj.size()<<endl;
            cout<<"Class Definition - "<<classDec.size()<<endl;
            cout<<"Constructor Definition - "<<constDec.size()<<endl;
            cout<<"Inherited Class Definition -
"<<inherClass.size()<<endl;
            cout<<"Operator Overloaded function Definition -
"<<opOverload.size()<<endl;
            for(auto it = className.begin();it!=className.end();it++){
                // cout<<*it<<endl;
            }
            // cout << keyword_return<< " " << keyword_class  << " "  <<
keyword_operator << endl;
            exit(1);
        }
        current_line++;
        str = trim(str);
        // cout<<str<<endl;
        i=0;
    }
}

int checkClass(string &func){
    while(str[i]!='{' && str[i]!=':' && str[i]!=' ' && str[i]!=';' &&
i!=str.size()){
        func += str[i];
        i++;
    }
    if(str[i]==';')return 2;
    nextLine();
    while(str[i] != '{' && str[i] != ':' && str[i] != ';'){
        i++;
        nextLine();
    }
    if(str[i] == '{'){
        return 0;
    }else if(str[i] == ':'){
        return 1;
    }else{
        return 2;
    }
}

void detectClass(){
    string func;
    i++;
    int line = current_line;
    int val = checkClass(func);
    className.insert(func);
    if(val==0){
        classDec.insert(line);
        i++;
        nextLine();
        insideClass(func);
    }else if(val==1){
        i++;
        nextLine();
        while(str[i]!='{' && str[i]!=';'){
            i++;
            nextLine();
        }
        if(str[i]=='{'){
            classDec.insert(line);
            inherClass.insert(line);
            i++;
            nextLine();
            insideClass(func);
        }
    }
    i++;
    nextLine();
}

void comment(){
    i++;
    if(str[i]=='/'){
        i = str.size();
        nextLine();
    }else if(str[i]=='*'){
        i++;
        nextLine();
        while(i<str.size()-1 && (str[i]!='*' || str[i+1]!='/')){
            i++;
            nextLine();
        }
    }
}

void insideClass(string name){
    int ob = 1;
    while(ob!=0){
        // cout<<ob<<endl;
        // cout<<"inside"<<endl;
        string word;
        while(i!=str.size() && isalnum(str[i])){
            word += str[i];
            i++;
        }
        nextLine();
        // cout<<word<<endl;
        bool flag = false;
        if(word==keyword_class || word == "class"){
            detectClass();
        }
        else if(word==keyword_operator || word == "operator"){
            int line = current_line;
            while(str[i]!='{' && str[i]!=';'){
                i++;
                nextLine();
            }
            if(str[i]=='{'){
                opOverload.insert(line);
            }
        }else if(className.find(word)!=className.end()){
            if(word!=name){
                while(str[i]==' '){
                    i++;
                    nextLine();
                    // flag = true;
                }
                if(isalnum(str[i])){
                    obj.insert(current_line);
                    while(str[i]!=';'){
                        i++;
                        nextLine();
                    }
                }
                nextLine();
            }
        }else if(word==keyword_return || word=="return"){
            while(str[i]!=';'){
                i++;
                nextLine();
            }
        }

        while(str[i]==' '){
            i++;
            nextLine();
            // flag = true;
        }
        if(str[i]=='('){
            if(word==name){
                int line = current_line;
                while(str[i]!='{' && str[i]!=';'){
                    i++;
                    nextLine();
                }
                if(str[i]=='{'){
                    ob++;
                    constDec.insert(line);
                }
            }
            i++;
            nextLine();
        }else if(str[i]=='{'){
            ob++;
            i++;
            nextLine();
        }else if(str[i]=='}'){
            ob--;
            i++;
            nextLine();
        }else if(str[i]=='~'){
            while(str[i]!='('){
                i++;
                nextLine();
            }
            i++;
            nextLine();
        }else{
            if(!isalnum(str[i])){
                i++;
                nextLine();
            }
        }
        // if(!flag){

        // }
    }
    while(str[i]!=';'){
        i++;
        nextLine();
    }
    return;
}

int main(){
    nextLine();
      while(1)
      {
        string word;
        while(i!=str.size() && isalnum(str[i])){
            word += str[i];
            i++;
        }
        nextLine();
        if(word==keyword_class || word=="class"){
            detectClass();
        }else if(className.find(word)!=className.end()){
            while(str[i]==' ' || str[i]=='*'){
                i++;
                nextLine();
            }
            if(str[i]=='('){
                constDec.insert(current_line);
                while(str[i]!=')'){
                    i++;
                    nextLine();
                }
                i++;
                nextLine();
            }
            else if(i!=str.size() && isalnum(str[i])){
                int j = i;
                while(j<str.size() && (str[j]!='{' && str[j]!=';')){
                    j++;
                }
                if(j!=str.size() && str[j]!='{'){
                    obj.insert(current_line);
                    while(str[i]!=';'){
```

```
            i++;
            nextLine();
          }
        }
      }
      nextLine();
    }else if(word==keyword_operator || word == "operator"){
      // cout<<"test"<<endl;
      int line = current_line;
      while(str[i]!='{' && str[i]!=';'){
        i++;
        nextLine();
      }
      if(str[i]=='{')
        opOverload.insert(line);
    }else if(word==keyword_return || word=="return"){
      while(str[i]!=';'){
        i++;
        nextLine();
      }
    }
    else{
      if (str[i]=='~')
      {
        while(str[i]!=';'){
          i++;
          nextLine();
        }
      }
      if(!isalnum(str[i])){
        i++;
        nextLine();
      }
    }
   }
   return 0;
}
```