CONTENTS

1. INTRODUCTION

1.1 Problem Statement

Cancer is the most common disease that occurs in people. Doctor’s diagnosis is reliable, but the procedure takes lots of time, and efforts to diagnose patients. Due to the increase in the number of people suffering from different types of cancers, it has made much more difficult for the oncologist to diagnose the cancer. These routines can be automated. It could save lots of time and could help to diagnose more accurate. Besides, using computerized means there are good opportunity to store information with diagnostic information in order to use it for further investigations or creation of new methods of diagnosis. In many cases , decision making involves remembering a large amount of data to diagnose a cancer. To overcome such difficulties, researchers are applying the same approach with expert system of an *Artificial Intelligence* to diagnose cancer.

1.2 Solution

In this project, the expert system of Artificial Intelligence is developed to overcome the difficulties of the problem mentioned above. The information of the symptoms of cancer type and corresponding treatment are stored in the knowledge base. This project uses the inference engine techniques such as ***backward chaining*** used to detect the cancer type and the ***forward chaining*** is used to provide the treatment. In backward chaining, when the user inputs the conclusion, the system will backtrack and will scan all the clauses in the IF statement of conclusion variable rule and will try to satisfy each clause. If a clause contains conclusion of different rule, then it moves to that rule containing the conclusion and satisfies all clauses belongs to that rule. It goes until it backtracks completely. The forward chaining proceeds in the forward way by checking the conditions until it reaches the conclusion. This system helps oncologists in diagnosing the cancer and to recommend he treatment.

2. CONTRIBUTIONS

2.1 Team Members

1. *Doti SandhyaRani*

I worked on collecting the cancer symptoms and treatment for each type of cancer, developed the decision tree and then created rules for forward chaining, developed algorithm for forward chaining and created data structures and developed forward chaining code in c++, performed the regression testing for forward chaining code, and rigorous testing of forward, backward and integrated code.

2. *Akshay Chandrachood*

Akshay worked on creating decision tree from the collected symptoms, then generated rules and developed backward chaining code in c++ with interface and data logging facility. Integrated the code for backward and forward chaining. Tested the integrated code.

3. ANALYSIS OF THE PROBLEM

3.1 Domain

The system domain: The Oncologist and patients at the Cancer Center.

3.2 Goal

The goal is to design an intelligent system to help the oncologist at cancer center to better diagnose patients by providing them with the symptoms based on the disease.

3.3 Problem with the Existing Code

We reviewed the code provided by the professor. It is inefficient and erroneous , because of the usage of goto statements and global variables, though it is logically correct. There is no separation between the knowledge base and the algorithm.

3.4 Proposal

To implement an intelligent expert system , our system should achieves the same functionality as per the provided code but with more user friendliness, more reliability and more efficiency. We have provided the user the facility to detect the specific type of the cancer and suggesting the treatment with better user friendliness. Once the program is running you can search for any number of cancers and get the treatments without running the system again. We have separated the algorithm from the knowledge base which increases the readability of our code.

4. KNOWLEDGE BASE DESIGN

4.1 Introduction

The knowledge base represents fact base and rule base. The knowledge base includes the following cancer types:

1. Wheezing
2. Large Cell Carcinoma
3. Squanors Cell Carcinoma
4. Large Cell Neuroma
5. Adeno Carcinoma
6. Nodular
7. Acral Lentigious
8. Lentigo
9. Super Facial Spreading
10. Leukamia Stage1
11. Acute Myclog
12. Chronic Myclog
13. Chronic Lymph
14. Acute Lymph
15. Acoustic Neuroma
16. CNS Lymphoma

17. Medullaoblastoma

18. Pituitary Tumours

19. Enolangio Carcinoma

20. Hepatoblastoma

21. Metastasis

22. Angiosarcoma

We gathered the data about symptoms of each of these cancer types, and corresponding treatments.

4.2 Decision Tree

- The decision tree is so named because it branches off just like a tree, and at the very end of each branch or system of branches is a conclusion.

- The decision tree has three symbols as follows:

Conclusion – ▬

Decision node – ⬬

arc or branch – →

The decision tree has circles and rectangles called "***nodes***" . The arrow lines that connect these nodes are known as **"*arcs*"** or "***branches***." The circles which contain questions are "***decision nodes*.”**

The rectangular shapes contain the goals of the diagram, and they signify conclusions. The arrow lines designate the direction of the diagram. Many of the nodes have branches leaving them, providing pathways to other nodes.

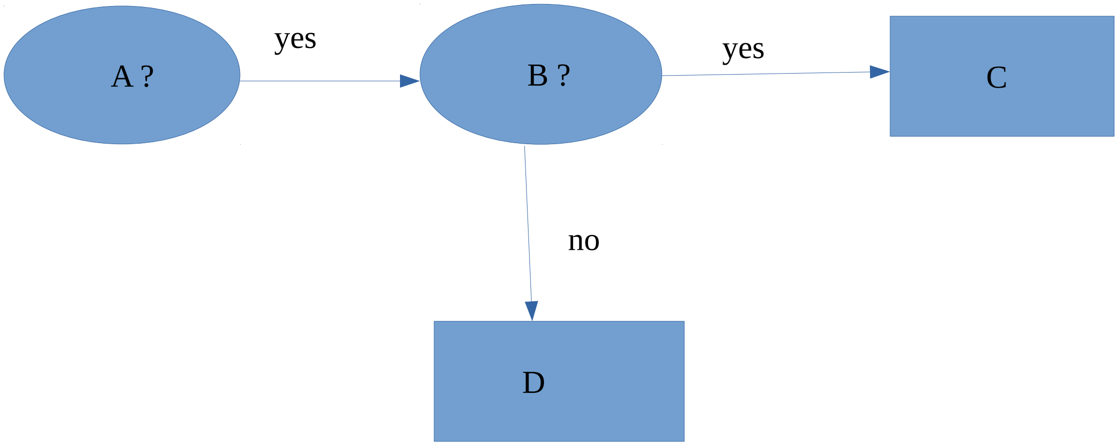
4.3 Converting tree to rules

The rules can be generated from the decision tree. Each rule has IF part and THEN part.

Example:

IF condition
THEN conclusion

If the IF condition evaluates to *true* then the THEN part of the conclusion gets executed.



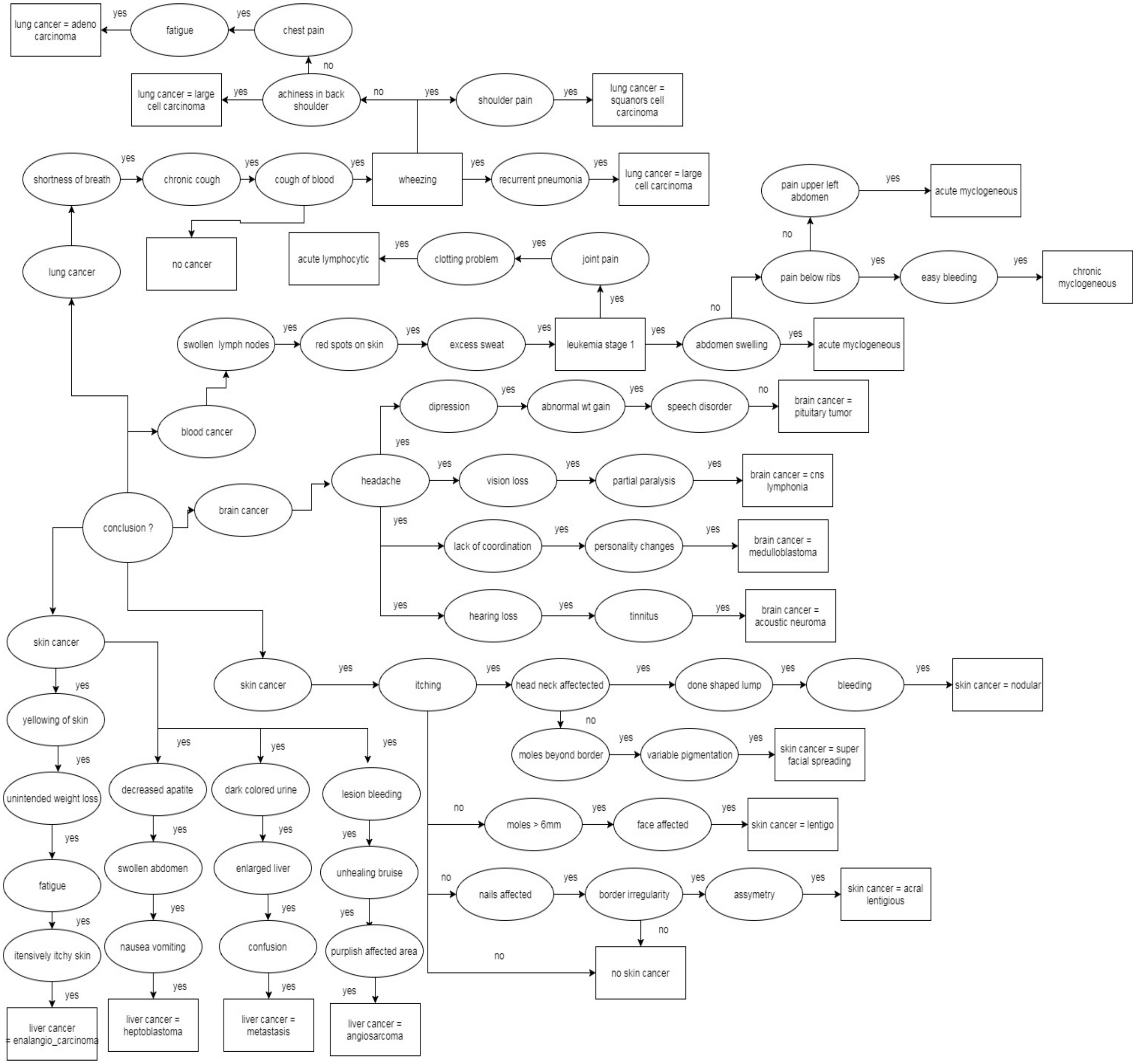
This will generate the rules like this,

1) IF A == yes and B == yes THEN C = yes

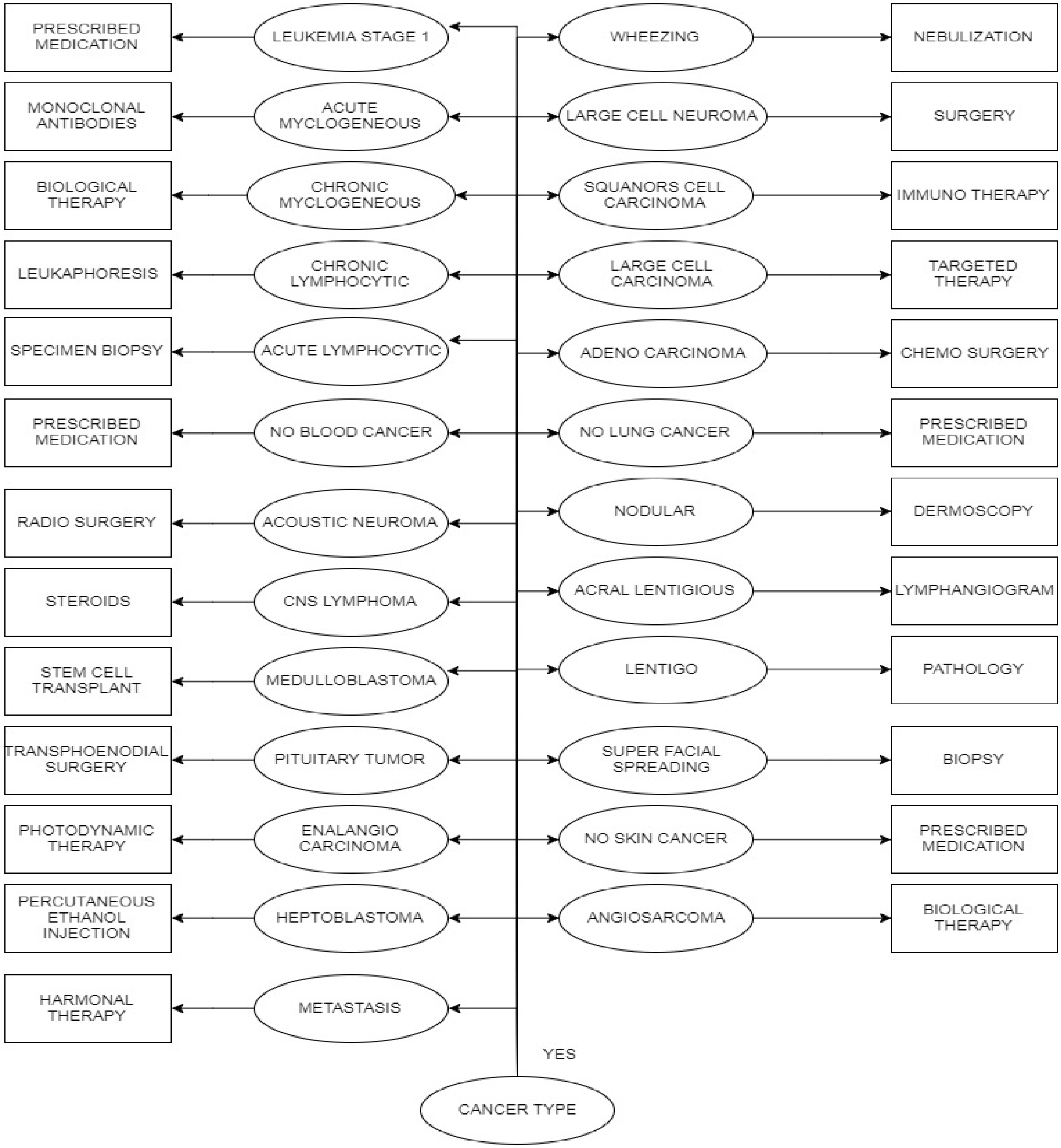
2) IF A == yes and B == no THEN D = yes

5. INFERENCE ENGINE

5.1 Decision Tree for backward chaining



5.2 Decision tree for forward chaining



5.3 Rules

5.3.1 Backward Chaining rules

10 IF Short_of_breath == yes AND Cough_of_blood == yes and Chronic_cough == yes
THEN lung_cancer = wheezing.

20 IF wheezing == yes and Recurrent_pneumonia == yes
THEN lung_cancer == large_cell_neuroma

30 IF wheezing == yes and shouder_pain == yes
THEN lung_cancer = squanors_cell_carcinoma

40 IF wheezing == yes and achiness_in_back_shoulder == yes
THEN lung_cancer = large_cell_carcinoma

50 IF wheezing == no and achinees_in_back_shoulder == no and chest_pain == yes and fatigue == yes THEN lung_cancer = adeno_carcinoma

60 IF short_of_breath == yes and chronic_cough = yes and cough_of_blood = no
THEN lung_cancer = No_lung_cancer

70 IF itching == yes and affected_area_head == yes and done_shaped_lump == yes and bleeding == yes THEN skin_cancer = nodular

80 IF itching == no and affected_area_nails == yes and border_irregularity == yes and assymetry == yes THEN skin_cancer = acral_lentigious

90 IF itching == no and moles == yes and affected_area_face == yes and mole_size>6mm == yes
THEN skin_cancer = lentigo

100 IF itching == yes and affected_area_head == no and moles_spread_beyond_border == yes and variable_pigmentation == yes THEN skin_cancer == super_facial_spreading

110 IF itching == yes and affected_area_nails = no THEN skin_cancer = no_skin_cancer

120 IF swollen_lymph_nodes == yes and red_spots == yes and excess_sweat == yes
THEN blood_ cancer = leukamia_stage1

130 IF leukamia_stage1 == yes and swelling_abdomen == yes
THEN blood_cancer = acute_my_clog

140 IF leukamia_stage1 = yes and swelling_abdomen = no and pain_in_below_ribs = yes and easy_bleeding = yes THEN blood_cancer = chronic_myclog

150 IF leukamia_stage1 = yes and swelling_abdomen = no and pain_below_ribs = no and pain_in_upper_left_abdomen = yes THEN blood_cancer = chronic_lymph

160 IF leukamia_stage1 = yes and clotting_problem = yes and joint_pain = yes
THEN blood_cancer = acute_lymph

170 IF swollen_lymph_nodes = no and red_spots = no
THEN blood_cancer = no_blood_cancer

180 IF headache == yes and vision_loss == no and hearing_loss == yes and tinnitus == yes
THEN brain_cancer = acoustic_carcinoma

190 IF headache == yes and vision_loss == yes and partial_paralysis == yes
THEN brain_cancer = cns_lymphonia

200 IF headache == yes and lack_of_coordination == yes and personality_changes == yes
THEN brain_cancer = medulloblastoma

210 IF headache == yes and speech_disorder == no and depression == yes and abnormal_wt_gain == yes THEN brain_cancer = pituitary _tumor

220 IF yellow_skin == yes and unintended_wt_loss == yes and fatigue == yes and intensity_itching_skin == yes THEN liver_cancer = enalangio_carcinoma

230 IF decreased_appetite == yes and swollen_abdomen == yes and nausea == yes THEN liver_cancer = hepatoblastoma

240 IF dark_colored_urine == yes and enlarged_liver == yes and confusion == yes and pian_in_upperright_abdomen == yes THEN liver_cancer == metastasis

250 IF lesion_bleeding == yes and unhealing_bruise == yes and purplish_effected_area == yes THEN liver_cancer = angiosarcoma

5.3.2 Forward Chaining Rules

```
10. IF cancer_type == wheezing
        THEN treatment = mebulization

20. IF cancer_type == large_cell_neuroma
        THEN treatment = surgery

30. IF cancer_type == squanors_cell_carcinoma
        THEN treatment = immuno_therapy

40. IF cancer_type == large_cell_carcinoma
        THEN treatment = targeted_therapy

50. IF cancer_type == adeno_carcinoma
        THEN treatment = chemotherapy

60. IF cancer_type == no_lung_cancer
        THEN treatment = prescribed medication

70. IF cancer_type == nodular
        THEN treatment = dermoscopy

80. IF cancer_type == acral_lentigious
        THEN treatment = lymphangiogram

90. IF cancer_type == lentigo
        THEN treatment = pathology

100. IF cancer_type == super_facial_spreading
        THEN treatment = biopsy

110. IF cancer_type == no_skin_cancer
        THEN treatment =  prescribed medication

120. IF cancer_type == leukamia_stage1
        THEN treatment =  prescribed medication
```

```
130.  IF cancer_type == acute_myclogenous
          THEN treatment = monoclonal antibodies

140. IF cancer_type == chronic_myclogenous
          THEN treatment = biological_therapy

150. IF cancer_type == chronic_lymphocytic
          THEN treatment = leukaphoresis

160. IF cancer_type == acute_lymphocytic
          THEN treatment = specimen_biopsy

170. IF cancer_type == no_blood_cancer
          THEN treatment = prescribed_medication

180. IF cancer_type == acoustic_neuroma
          THEN treatment = radio_surgery

190. IF cancer_type == cns_lymphoma
          THEN treatment = steriods

200. IF cancer_type == medulloblastoma
          THEN treatment = stemcell_transplant

210. IF cancer_type == pituitary_tumour
          THEN treatment = transphenodial_surgery

220. IF cancer_type == enalangio_carcinoma
          THEN treatment = photodynamic_therapy

230. IF cancer_type == heptoblastoma
          THEN treatment = percutaneous_ethanol_injection

240. IF cancer_type == liver_metastasis
          THEN treatment = hormonal_therapy

250. IF cancer_type == angio_sarcoma
          THEN treatment = biological_therapy
```

5.4 Backward Chaining

5.4.1 Introduction

Backward Chaining is an inference method that can be described as working backward from the goals. It starts with a list of conclusions and works backwards from the IF part to see if there is data available that will support any of these conclusions. An inference engine using backward chaining would search the inference rules until it finds one which has a conclusion that matches the desired goal. If the if clause of that rule is not known, then it is added to the list of goals.

Example:

1. If x == croaks and x == flies Then x = frog
2. If x == frog Then x is green

5.4.2 Algorithm

Step 1. Get the conclusion from the user.

Step 2. Scan the conclusion list for the first instance of the conclusion's name. If found, place the rule on the conclusion stack using the rule number and a (1) to represent the clause number. If not found, notify the user that an answer cannot be found..

Step 3. Instantiate the IF clause (i.e., each condition variable) of the statement.

Step 4. If one of the IF clause variables is not instantiated, as indicated by the variable list, and is not a conclusion variable, that is, not on the conclusion list, ask the user to enter a value.

Step 5. If one of the clauses is a conclusion variable, place the conclusion variable's rule number on the top of the stack and go back to step 3.

Step 6. If the statement on top of the stack cannot be instantiated using the present IF-THEN statement, remove the unit from the top of the stack and search the conclusion list for another instance of that conclusion variable's name.

Step 7. If such a statement is found, go back to step 3.

Step8. If there are no more conclusions left on the conclusion stack with that name, the rule for the previous conclusion is false. If there is no previous conclusion, then notify the user that an answer cannot be found. If there is a previous conclusion, go back to step 6.

Step9. If the rule on top of the stack can be instantiated, remove it from the stack. If another conclusion variable is underneath, increment the clause number, and for the remaining clauses go back to step 3. If no other conclusion variable is underneath, we have answered our question. The user can come to a conclusion

5.4.3 Data structures needed for backward chaining

- ***Conclusion list***

  Contains all the conclusions in the list. It has two columns , one is rule number and the other is the conclusion variable.

- ***Conclusion stack***

  It is used to keep track of which rule and which clause within that rule we are trying to reach.

- ***Clause Variable List***

  Contains clause variables for each of the rule.

- ***Variable List***

  Lists all the variables and its values. It has two columns one is variable name to store IF part of the knowledge base and the other column indicates whether variable is initialized or not.

5.4.4 Clause variable list

We allocate room for four variables for each rule.

Rule #	Clause Variable Name
1	short_breath
2	chronic_cough
3	cough_blood
4	
5	wheezing
6	recurrent_pneumonia
7	
8	
9	wheezing
10	shoulder_pain
11	
12	
13	wheezing
14	achiness_back_shoulder
15	
16	
17	wheezing
18	achiness_back_shoulder
19	chest_pain
20	fatigue
21	short_breath
22	chronic_cough
23	cough_blood
24	
25	itching
26	head_neck_affected
27	done_shaped_lump

28	bleeding
29	itching
30	nails_affected
31	border_irregularity
32	assymetry
33	itching
34	moles>6mm
35	face_affected
36	
37	itching
38	moles_beyond_border
39	head_neck_affected
40	variable_pigmentation
41	itching
42	border_irregularity
43	
44	
45	swollen_lymph_nodes
46	red_spots_on_skin
47	excess_sweat
48	
49	leukemia_stage1
50	abdomen_swelling
51	
52	
53	leukemia_stage1
54	abdomen_swelling
55	pain_below_ribs
56	easy_bleeding
57	leukemia_stage1
58	abdomen_swelling

59	pain_below_ribs
60	pain_upper_left_abdomen
61	leukemia_stage1
62	joint_pain
63	clotting_problem
64	
65	leukemia_stage1
66	
67	
68	
69	head_ache
70	vision_loss
71	hearing_loss
72	tinnitus
73	head_ache
74	vision_loss
75	partial_paralysis
76	
77	head_ache
78	lack_of_coordination
79	personality_changes
80	
81	head_ache
82	depression
83	abnormal_wt_gain
84	speech_disorder
85	yellow_skin
86	weight_loss
87	deep_fatigue
88	itensively_itchy_skin
89	decreased_apatite

90	swollen_abdomen
91	nausea_vomiting
92	
93	dark_colored_urine
94	enlarged_liver
95	confusion
96	
97	lesion_bleeding
98	unhealing_bruise
99	purplish_affected_area
100	

5.4.5 Conclusion List

10	wheezing
20	lung_cancer
30	lung_cancer
40	lung_cancer
50	lung_cancer
60	lung_cancer
70	skin_cancer
80	skin_cancer
90	skin_cancer
100	skin_cancer
110	skin_cancer
120	leukemia_stage1
130	blood_cancer
140	blood_cancer
150	blood_cancer
160	blood_cancer

170	blood_cancer
180	brain_cancer
190	brain_cancer
200	brain_cancer
210	brain_cancer
220	liver_cancer
230	liver_cancer
240	liver_cancer
250	liver_cancer

5.4.6 Variable List

1	short_breath
2	chronic_cough
3	cough_blood
4	recurrent_pneunomia
5	shoulder_pain
6	achiness_back_shoulder
7	chest_pain
8	fatigue
9	itching
10	head_neck_affected
11	done_shaped_lump
12	bleeding
13	nails_affected
14	border_irregularity
15	assymetry
16	moles>6mm

17	face_affected
18	moles_beyond_border
19	variable_pigmentation
20	swollen_lymph_nodes
21	red_spots_on_skin
22	excess_sweat
23	abdomen_swelling
24	joint_pain
25	clotting_problem
26	pain_below_ribs
27	easy_bleeding
28	pain_upper_left_abdomen
29	head_ache
30	vision_loss
31	hearing_loss
32	tinnitus
33	partial_paralysis
34	lack_of_coordination
35	personality_changes
36	depression
37	abnormal_wt_gain
38	speech_disorder
39	yellow_skin
40	weight_loss
41	deep_fatigue
42	itensively_itchy_skin
43	decreased_apatite
44	swollen_abdomen
45	nausea_vomiting
46	
47	dark_colored_urine

48	enlarged_liver
49	confusion
50	
51	lesion_bleeding
52	unhealing_bruise
53	purplish_affected_area
54	

5.5 Forward Chaining

5.5.1 Introduction

Forward chaining starts with the available data and uses inference rules to extract more data until a goal is reached. An inference engine using forward chaining searches the inference rules until it finds one where the **If** clause is known to be true. When such a rule is found, the engine can conclude, or infer, the **Then** clause, resulting in the addition of new information to its data. The name "forward chaining" comes from the fact that the inference engine starts with the data and reasons its way to the answer.

Example:

1. If x chirps and x sings THEN x is a canary

2. If x is a canary THEN x is yellow

- We begin with the condition i.e; x chirps and x sings
- We execute rule1 we get the conclusion x is a canary.
- We scan if any rules have x as a canary
- We go to rule2 , in which we get the conclusion x is yellow.

5.5.2 Algorithm

step 1. The condition is identified.

Step 2. The condition variable is placed on the conclusion variable queue and its value is marked on the variable list.

Step 3. The clause variable list is searched for the variable whose name is the same as the one in the front of the queue. If found, the rule number and a 1 are placed into the clause variable pointer. If not found, go to step 6.

step 4. Each variable in the IF clause of the rule that is not already instantiated is now instantiated. The variables are in the clause variable list. If all the clauses are true, the THEN part is invoked.

Step 5. The instantiated THEN part of the variable is placed in the *back* of the conclusion variable queue.

Step 6. When there are no more IF statements containing the variable that is at the *front* of the conclusion variable queue, that variable is removed.

Step 7. If there are no more variables on the conclusion variable queue, end the session. If there are more variables, go to step 3.

5.5.3 Data structures needed for forward chaining

- ***Clause Variable List***

  Contains clause variables for each rule.
- ***Variable List***

  Lists all the variables in the rules and its values.
- ***Conclusion Variable Queue***

  It contains all the variables that are needed to be initialized to reach the conclusion.
- ***Clause variable pointer***

  It keeps track of the clause variable that we are examining. It has rule number and clause number.

5.5.4 Variable List

1. CANCER_TYPE

5.5.5 Clause variable list

Each rule is allocated with room space of two

1	CANCER_TYPE
2	
3	CANCER_TYPE
4	
5	CANCER_TYPE
6	
7	CANCER_TYPE
8	
9	CANCER_TYPE
10	
11	CANCER_TYPE
12	
13	CANCER_TYPE
14	
15	CANCER_TYPE
16	
17	CANCER_TYPE
18	
19	CANCER_TYPE
20	
21	CANCER_TYPE
22	
23	CANCER_TYPE
24	

25	CANCER_TYPE
26	
27	CANCER_TYPE
28	
29	CANCER_TYPE
30	
31	CANCER_TYPE
32	
33	CANCER_TYPE
34	
35	CANCER_TYPE
36	
37	CANCER_TYPE
38	
39	CANCER_TYPE
40	
41	CANCER_TYPE
42	
43	CANCER_TYPE
44	
45	CANCER_TYPE
46	
47	CANCER_TYPE
48	
49	CANCER_TYPE
50	

6. CODE

6.1 Backward Chaining

main.cpp

```
#include <iostream>
#include <stack>
#include<vector>
#include <string>
#include <string.h>
#include <fstream>
#include "data_structures.h"
#include "forward.cpp"

#define var_list_size 54
#define clause_list_size 101
#define conc_list_size 25


using namespace std;

void initialize_lists();
int conclusion_search(int,string);
void update_variable_list(int rule);
void evaluate_then_part(int rule);
void check_var_list(string var_to_check);
void iterate(int);
```

```
ConclusionList conc_list[conc_list_size];

VariableList var_list[var_list_size];
ClauseVarList clause_vars[clause_list_size];
bool rule_found();

stack<int> cn_stack;
stack<int> sn_stack;

int sn;
int cn,count;
int pos;
int var_in_clause_list;
string conclusion;
int case_no=-1;


void backward_chaining(int sn, string conclusion);
void interface();
void write_file();

//----------------------------- Main ------------------------------------------------

int main()
{

   char answer;

   write_file();
```

```
//initialize variable list, conclusion variable list & clause variable list
initialize_lists();
interface();

ofstream fout;
fout.open("data_log.txt", ios::app);



do{
int con_no=0;
cout<<endl<<"Please Enter The Conclusion: ";
cin>>conclusion;

initialize_lists();

pos=0;

   sn = conclusion_search(pos,conclusion);
   if(sn==-1){
      cout<<"Conclusion Not Found.."<<endl;
      return 0;
   }

   else
      backward_chaining(sn, conclusion);



   while(conc_list[pos].get_conclusion_value()=="" && pos<conc_list_size){
      while(!sn_stack.empty()){
         sn_stack.pop();
      }
      pos++;
      sn = conclusion_search(pos,conclusion);
      backward_chaining(sn, conclusion);
   }

   cout<<endl<<"******************************** REPORT
****************************"<<endl<<endl;

while(!sn_stack.empty()){

   switch(sn_stack.top()){
      case 10: con_no = 0; break;
      case 20: con_no = 1; break;
      case 30: con_no = 2; break;
      case 40: con_no = 3; break;
      case 50: con_no = 4; break;
      case 60: con_no = 5; break;
      case 70: con_no = 6; break;
      case 80: con_no = 7; break;
      case 90: con_no = 8; break;
      case 100: con_no = 9; break;
      case 110: con_no = 10; break;
      case 120: con_no = 11; break;
      case 130: con_no = 12; break;
      case 140: con_no = 13; break;
      case 150: con_no = 14; break;
```

```
            case 160: con_no = 15; break;
            case 170: con_no = 16; break;
            case 180: con_no = 17; break;
            case 190: con_no = 18; break;
            case 200: con_no = 19; break;
            case 210: con_no = 20; break;
            case 220: con_no = 21; break;
            case 230: con_no = 22; break;
            case 240: con_no = 23; break;
            case 250: con_no = 24; break;
        }
        if(conc_list[con_no].get_conclusion_value()!=""){

            cout<<endl<<endl<<"The Conclusion is: "<<endl;
            cout<<"Evaluated Rule: "<<sn_stack.top()<<endl;
            cout<<"Result: "<<conc_list[con_no].get_conclusion()<<" = "<<
conc_list[con_no].get_conclusion_value()<<endl;
            fout<<"Result: "<<conc_list[con_no].get_conclusion()<<" = "<<
conc_list[con_no].get_conclusion_value()<<endl;
        }
        sn_stack.pop(); cn_stack.pop();
     }

cout<<endl<<"************************************************************"<<endl;

   if(con_no==0)
       Forward_Chaining d1(25);
   else
       Forward_Chaining d1(con_no);
```

```
  /*  while(!sn_stack.empty()){
        sn_stack.pop();
   }
   while(!cn_stack.empty()){
        sn_stack.pop();
   }*/
   //con_no=0;
   cout<<"Do you want to continue <y/n> : ";
   cin>>answer;
   }while(answer=='y' || answer=='Y');

   return 0;
}




void backward_chaining(int sn, string conclusion)
{
   sn_stack.push(sn);
   cn = 4 * (sn/10-1) + 1;
   cn_stack.push(cn);

   int var_in_clause_list = 1;
   do{
        string var_to_check = clause_vars[cn].get_clause_vars();
        int new_sn = conclusion_search(pos, var_to_check);

        if(new_sn!=-1){
            pos++;
            backward_chaining(new_sn, var_to_check);
        }
        check_var_list(var_to_check);
        cn++;
        var_in_clause_list++;
    }while(var_in_clause_list<5 && clause_vars[cn].get_clause_vars()!="");

    evaluate_then_part(sn);

}

void check_var_list(string var_to_check)
{
    for(int i=0;i<var_list_size;i++)
    {
        if(var_to_check == var_list[i].getVarName())
        {
            if(!var_list[i].getStatus())
            {
                string question;
                cout<<var_list[i].getVarName()<<" <yes/no>: ";
                cin>>question;
                var_list[i].set(question);
                var_list[i].setStatus(1);
            }
        }
    }
}
```

```
int conclusion_search(int pos, string var)
{
   for(int i=pos;i<conc_list_size;i++)
   {
      if(var == conc_list[i].get_conclusion())
      {
         return conc_list[i].get_rule();
      }
   }
   return -1;
}


void evaluate_then_part(int rule)
{
      switch(rule)
      {
         case 10:    if(var_list[0].getValue() == "yes" && var_list[1].getValue() == "yes" &&
var_list[2].getValue() == "yes")
                     {
                        conc_list[0].set_value("yes");
                        case_no = 0;
                     }else{conc_list[0].set_value("no");}
                     break;

         case 20:   if(conc_list[0].get_conclusion_value() == "yes" &&  var_list[3].getValue() == "yes")
                     {
                        conc_list[1].set_value("large_cell_neuroma");
```

```
                case_no = 1;
            }
            break;

    case 30:    if(conc_list[0].get_conclusion_value() == "yes" && var_list[4].getValue() == "yes")
            {
                conc_list[2].set_value("squanors_cell_carcinoma");
                case_no = 2;
            }
             break;

     case 40:    if(conc_list[0].get_conclusion_value() == "no" &&  var_list[5].getValue() == "yes")
            {
                conc_list[3].set_value("large_cell_carcinoma");
                case_no = 3;
            }
            break;

     case 50:    if(conc_list[0].get_conclusion_value() == "no" && var_list[5].getValue() == "no"
&& var_list[6].getValue() == "yes" && var_list[7].getValue() == "yes")
            {
                conc_list[4].set_value("adeno_carcinoma");
                case_no = 4;
            }
            break;

     case 60:    if(var_list[0].getValue() == "yes" && var_list[1].getValue() == "yes" &&
var_list[2].getValue() == "no")
            {
                   conc_list[5].set_value("no_lung_cancer");
                   case_no = 5;
                }
                break;
        case 70:    if(var_list[8].getValue() == "yes" && var_list[9].getValue() == "yes" &&
var_list[10].getValue() == "yes" && var_list[11].getValue() == "yes")
                {
                   conc_list[6].set_value("nodular");
                   case_no = 6;
                }
                break;

        case 80:   if(var_list[8].getValue() == "no" && var_list[12].getValue() == "yes" &&
var_list[13].getValue() == "yes" && var_list[14].getValue() == "yes")
                {
                   conc_list[7].set_value("acral_lentigious");
                   case_no = 7;
                }
                break;

        case 90:   if(var_list[8].getValue() == "no" && var_list[15].getValue() == "yes" &&
var_list[16].getValue() == "yes")
                {
                   conc_list[8].set_value("lentigo");
                   case_no = 8;
                }
                break;

         case 100:  if(var_list[8].getValue() == "yes" && var_list[9].getValue() == "no"
                  && var_list[17].getValue() == "yes" && var_list[18].getValue() == "yes")
                {
                   conc_list[9].set_value("super_facial_spreading");
                   case_no = 9;
                }
                break;

        case 110:   if(var_list[8].getValue() == "no" && var_list[13].getValue() == "no")
                {
                   conc_list[10].set_value("no_skin_cancer");
                   case_no =10;
                }
                break;

        case 120:   if(var_list[19].getValue() == "yes" && var_list[20].getValue() == "yes" &&
var_list[21].getValue() == "yes")
                {
                   conc_list[11].set_value("yes");
                   case_no =11;
                }else{conc_list[11].set_value("no");}
                break;

         case 130:  if(conc_list[11].get_conclusion_value() == "yes" && var_list[22].getValue() ==
"yes" )
                {
                   conc_list[12].set_value("acute_myclogenous");
                   case_no =12;
                }
                break;
        case 140:  if(conc_list[11].get_conclusion_value() == "yes" && var_list[22].getValue() ==
"no" && var_list[25].getValue() == "yes" && var_list[26].getValue() == "yes")
              {
                conc_list[13].set_value("chronic_myclogenous");
                case_no =13;
              }
              break;

       case 150:   if(conc_list[11].get_conclusion_value() == "yes" && var_list[22].getValue() ==
"no" && var_list[25].getValue() == "no" && var_list[27].getValue() == "yes")
              {
                conc_list[14].set_value("chronic_lymphocytic");
                case_no =14;
              }
              break;

        case 160:  if(conc_list[11].get_conclusion_value() == "yes" && var_list[23].getValue() ==
"yes" && var_list[24].getValue() == "yes")
              {
                conc_list[15].set_value("acute_lymphocytic");
                case_no =15;
              }
              break;

        case 170:  if(conc_list[11].get_conclusion_value() == "no")
              {
                conc_list[16].set_value("No Blood Cancer");
                case_no =16;
```

```
                 }
                 break;

            case 180:  if(var_list[28].getValue() == "yes" && var_list[29].getValue() == "no" &&
var_list[30].getValue() == "yes" && var_list[31].getValue() == "yes")
                 {
                    conc_list[17].set_value("acoustic_neuroma");
                    case_no =17;
                 }
                 break;

           case 190:   if(var_list[28].getValue() == "yes" && var_list[29].getValue() == "yes" &&
var_list[32].getValue() == "yes")
                 {
                    conc_list[18].set_value("cns_lymphoma");
                    case_no =18;
                 }
                 break;

            case 200:   if(var_list[28].getValue() == "yes" && var_list[33].getValue() == "yes" &&
var_list[34].getValue() == "yes")
                 {
                    conc_list[19].set_value("medulloblastoma");
                    case_no =19;
                 }
                 break;

           case 210:   if(var_list[28].getValue() == "yes" && var_list[35].getValue() == "yes" &&
var_list[36].getValue() == "yes" && var_list[37].getValue() == "no")
```

```
                        {
                           conc_list[20].set_value("pituitary_tumor");
                           case_no =20;
                        }
                        break;



            case 220:  if(var_list[38].getValue() == "yes" && var_list[39].getValue() == "yes" &&
var_list[40].getValue() == "yes" && var_list[41].getValue() == "yes")
                        {
                           conc_list[21].set_value("enalangio_carcinoma");
                           case_no =21;
                        }
                        break;

            case 230:   if(var_list[42].getValue() == "yes" && var_list[43].getValue() == "yes" &&
var_list[44].getValue() == "yes")
                        {
                           conc_list[22].set_value("heptoblastoma");
                           case_no =22;
                        }
                        break;

            case 240:  if(var_list[46].getValue() == "yes" && var_list[47].getValue() == "yes" &&
var_list[48].getValue() == "yes")
                        {
                           conc_list[23].set_value("metastasis");
                           case_no =23;
                        }
```

```
                  break;

            case 250:  if(var_list[50].getValue() == "yes" && var_list[51].getValue() == "yes" &&
var_list[52].getValue() == "yes")
                  {
                    conc_list[24].set_value("angiosarcoma");
                    case_no =24;
                  }
                  break;
        }
}




void initialize_lists()
{
    //cout<<"Initializing Variable List.."<<endl;
    //LUNG CANCER
    for(int i=0;i<var_list_size;i++){
        var_list[i].init(0,"","");
    }
    for(int i=0;i<conc_list_size;i++){
        conc_list[i].set_status(0);
        conc_list[i].set_value("");
    }

    var_list[0].init(0,"short_breath","");
    var_list[1].init(0,"chronic_cough","");
```

```
var_list[2].init(0,"cough_blood","");
var_list[3].init(0,"recurrent_pneunomia","");
var_list[4].init(0,"shoulder_pain","");
var_list[5].init(0,"achiness_back_shoulder","");
var_list[6].init(0,"chest_pain","");
var_list[7].init(0,"fatigue","");
//SKIN CANCER
var_list[8].init(0,"itching","");
var_list[9].init(0,"head_neck_affected","");
var_list[10].init(0,"done_shaped_lump","");
var_list[11].init(0,"bleeding","");
var_list[12].init(0,"nails_affected","");
var_list[13].init(0,"border_irregularity","");
var_list[14].init(0,"assymetry","");
var_list[15].init(0,"moles>6mm","");
var_list[16].init(0,"face_affected","");
var_list[17].init(0,"moles_beyond_border","");
var_list[18].init(0,"variable_pigmentation","");

var_list[19].init(0,"swollen_lymph_nodes","");
var_list[20].init(0,"red_spots_on_skin","");
var_list[21].init(0,"excess_sweat","");
var_list[22].init(0,"abdomen_swelling","");
var_list[23].init(0,"joint_pain","");
var_list[24].init(0,"clotting_problem","");
var_list[25].init(0,"pain_below_ribs","");
var_list[26].init(0,"easy_bleeding","");
var_list[27].init(0,"pain_upper_left_abdomen","");
```

```
var_list[28].init(0,"head_ache","");
var_list[29].init(0,"vision_loss","");
var_list[30].init(0,"hearing_loss","");
var_list[31].init(0,"tinnitus","");
var_list[32].init(0,"partial_paralysis","");
var_list[33].init(0,"lack_of_coordination","");
var_list[34].init(0,"personality_changes","");
var_list[35].init(0,"depression","");
var_list[36].init(0,"abnormal_wt_gain","");
var_list[37].init(0,"speech_disorder","");


var_list[38].init(0,"yellow_skin","");
var_list[39].init(0,"weight_loss","");
var_list[40].init(0,"deep_fatigue","");
var_list[41].init(0,"itensively_itchy_skin","");

var_list[42].init(0,"decreased_apatite","");
var_list[43].init(0,"swollen_abdomen","");
var_list[44].init(0,"nausea_vomiting","");
var_list[45].init(0,"","");

var_list[46].init(0,"dark_colored_urine","");
var_list[47].init(0,"enlarged_liver","");
var_list[48].init(0,"confusion","");
var_list[49].init(0,"","");

var_list[50].init(0,"lesion_bleeding","");
var_list[51].init(0,"unhealing_bruise","");
```

```
var_list[52].init(0,"purplish_affected_area","");
var_list[53].init(0,"","");



//------------------conclusion variable list------------------------
conc_list[0].set_rule(10,"wheezing");
conc_list[1].set_rule(20,"lung_cancer");  //large-cell-carninoma
conc_list[2].set_rule(30,"lung_cancer");
conc_list[3].set_rule(40,"lung_cancer");
conc_list[4].set_rule(50,"lung_cancer");
conc_list[5].set_rule(60,"lung_cancer");
 //Conclusions for Skin Cancer
conc_list[6].set_rule(70,"skin_cancer");
conc_list[7].set_rule(80,"skin_cancer");
conc_list[8].set_rule(90,"skin_cancer");
conc_list[9].set_rule(100,"skin_cancer");
conc_list[10].set_rule(110,"skin_cancer");

conc_list[11].set_rule(120,"leukemia_stage1");
conc_list[12].set_rule(130,"blood_cancer");
conc_list[13].set_rule(140,"blood_cancer");
conc_list[14].set_rule(150,"blood_cancer");
conc_list[15].set_rule(160,"blood_cancer");
conc_list[16].set_rule(170,"blood_cancer");

conc_list[17].set_rule(180,"brain_cancer");
conc_list[18].set_rule(190,"brain_cancer");
conc_list[19].set_rule(200,"brain_cancer");
conc_list[20].set_rule(210,"brain_cancer");
```

```
conc_list[21].set_rule(220,"liver_cancer");
conc_list[22].set_rule(230,"liver_cancer");
conc_list[23].set_rule(240,"liver_cancer");
conc_list[24].set_rule(250,"liver_cancer");



//-----------clause variable list--------------------
clause_vars[0].set_vars(0,"");
clause_vars[1].set_vars(1,"short_breath");
clause_vars[2].set_vars(2,"chronic_cough");
clause_vars[3].set_vars(3,"cough_blood");
clause_vars[4].set_vars(4,"");
clause_vars[5].set_vars(5,"wheezing");
clause_vars[6].set_vars(6,"recurrent_pneunomia");
clause_vars[7].set_vars(7,"");
clause_vars[8].set_vars(8,"");
clause_vars[9].set_vars(9,"wheezing");
clause_vars[10].set_vars(10,"shoulder_pain");
clause_vars[11].set_vars(11,"");
clause_vars[12].set_vars(12,"");
clause_vars[13].set_vars(13,"wheezing");
clause_vars[14].set_vars(14,"achiness_back_shoulder");
clause_vars[15].set_vars(15,"");
clause_vars[16].set_vars(16,"");
clause_vars[17].set_vars(17,"wheezing");
clause_vars[18].set_vars(18,"achiness_back_shoulder");
clause_vars[19].set_vars(19,"chest_pain");
clause_vars[20].set_vars(20,"fatigue");
```

```
clause_vars[21].set_vars(21,"short_breath");
clause_vars[22].set_vars(22,"chronic_cough");
clause_vars[23].set_vars(23,"cough_blood");
clause_vars[24].set_vars(24,"");

clause_vars[25].set_vars(25,"itching");
clause_vars[26].set_vars(26,"head_neck_affected");
clause_vars[27].set_vars(27,"done_shaped_lump");
clause_vars[28].set_vars(28,"bleeding");
clause_vars[29].set_vars(29,"itching");
clause_vars[30].set_vars(30,"nails_affected");
clause_vars[31].set_vars(31,"border_irregularity");
clause_vars[32].set_vars(32,"assymetry");
clause_vars[33].set_vars(33,"itching");
clause_vars[34].set_vars(34,"moles>6mm");
clause_vars[35].set_vars(35,"face_affected");
clause_vars[36].set_vars(36,"");
clause_vars[37].set_vars(37,"itching");
clause_vars[38].set_vars(38,"moles_beyond_border");
clause_vars[39].set_vars(39,"head_neck_affected");
clause_vars[40].set_vars(40,"variable_pigmentation");
clause_vars[41].set_vars(41,"itching");
clause_vars[42].set_vars(42,"border_irregularity");
clause_vars[43].set_vars(43,"");
clause_vars[44].set_vars(44,"");

clause_vars[45].set_vars(45,"swollen_lymph_nodes");
clause_vars[46].set_vars(46,"red_spots_on_skin");
clause_vars[47].set_vars(47,"excess_sweat");
```

```
clause_vars[48].set_vars(48,"");

clause_vars[49].set_vars(49,"leukemia_stage1");
clause_vars[50].set_vars(50,"abdomen_swelling");
clause_vars[51].set_vars(51,"");
clause_vars[52].set_vars(52,"");

clause_vars[53].set_vars(53,"leukemia_stage1");
clause_vars[54].set_vars(54,"abdomen_swelling");
clause_vars[55].set_vars(55,"pain_below_ribs");
clause_vars[56].set_vars(56,"easy_bleeding");

clause_vars[57].set_vars(57,"leukemia_stage1");
clause_vars[58].set_vars(58,"abdomen_swelling");
clause_vars[59].set_vars(59,"pain_below_ribs");
clause_vars[60].set_vars(60,"pain_upper_left_abdomen");
clause_vars[61].set_vars(61,"leukemia_stage1");
clause_vars[62].set_vars(62,"joint_pain");
clause_vars[63].set_vars(63,"clotting_problem"); //joint_pain
clause_vars[64].set_vars(64,"");
clause_vars[65].set_vars(65,"leukemia_stage1");
clause_vars[66].set_vars(66,"");
clause_vars[67].set_vars(67,"");
clause_vars[68].set_vars(68,"");

clause_vars[69].set_vars(69,"head_ache");
clause_vars[70].set_vars(70,"vision_loss");
clause_vars[71].set_vars(71,"hearing_loss");
clause_vars[72].set_vars(72,"tinnitus");
```

```
clause_vars[73].set_vars(73,"head_ache");
clause_vars[74].set_vars(74,"vision_loss");
clause_vars[75].set_vars(75,"partial_paralysis");
clause_vars[76].set_vars(76,"");
clause_vars[77].set_vars(77,"head_ache");
clause_vars[78].set_vars(78,"lack_of_coordination");
clause_vars[79].set_vars(79,"personality_changes");
clause_vars[80].set_vars(80,"");
clause_vars[81].set_vars(81,"head_ache");
clause_vars[82].set_vars(82,"depression");
clause_vars[83].set_vars(83,"abnormal_wt_gain");
clause_vars[84].set_vars(84,"speech_disorder");

clause_vars[85].set_vars(85,"yellow_skin");
clause_vars[86].set_vars(86,"weight_loss");
clause_vars[87].set_vars(87,"deep_fatigue");
clause_vars[88].set_vars(88,"itensively_itchy_skin");
clause_vars[89].set_vars(89,"decreased_apatite");
clause_vars[90].set_vars(90,"swollen_abdomen");
clause_vars[91].set_vars(91,"nausea_vomiting");
clause_vars[92].set_vars(92,"");
clause_vars[93].set_vars(93,"dark_colored_urine");
clause_vars[94].set_vars(94,"enlarged_liver");
clause_vars[95].set_vars(95,"confusion");
clause_vars[96].set_vars(96,"");

clause_vars[97].set_vars(97,"lesion_bleeding");
clause_vars[98].set_vars(98,"unhealing_bruise");
clause_vars[99].set_vars(99,"purplish_affected_area");
```

```
    clause_vars[100].set_vars(100,"");

}

void interface(){
    char user_ip1;
    string read_rules;

    ifstream fin;
    fin.open("rules.txt");



    cout<<endl<<"***EXPERT SYSTEM FOR SPECIFIC CANCER DETECTION AND
TREATMENT RECOMMENDATION***"<<endl<<endl;
    cout<<endl<<"Course: CS 5346\tARTIFICIAL INTELLIGENCE"<<endl<<endl;
    cout<<"Group Members: 1.Akshay Chandrachood\t2.SandhyaRani Doti"<<endl<<endl;

cout<<"****************************************RULES*************************************
**"<<endl<<endl;
    cout<<"Do you wish to read rules first? <y/n> : ";
    cin>>user_ip1;
    if(user_ip1=='Y' || user_ip1=='y'){
       while(getline(fin,read_rules)){
            cout<<read_rules<<endl<<endl;
         }
    }
    fin.close();
```

```
cout<<"****************************************************************************
**"<<endl<<endl;
   cout<<"=================================Conclusion
List=============================="<<endl<<endl;
   for(int i=0;i<conc_list_size;i=i+2){
      conc_list[i].print_rule();
      cout<<"\t\t\t ";
      if(i+1<conc_list_size)
      conc_list[i+1].print_rule();
      cout<<endl;
   }

cout<<endl<<"==============================================================
==========="<<endl<<endl;
}

void write_file(){
   ofstream fout;
   fout.open("data_log.txt");
   fout<<endl<<"***EXPERT SYSTEM FOR SPECIFIC CANCER DETECTION AND
TREATMENT RECOMMENDATION***"<<endl<<endl;
   fout<<endl<<"Course: CS 5346\tARTIFICIAL INTELLIGENCE"<<endl<<endl;
   fout<<"Group Members: 1.Akshay Chandrachood\t2.SandhyaRani Doti"<<endl<<endl;

fout<<"____________________________________________________________________________
_______"<<endl<<endl;
}
```

data_structures.h

```
#ifndef DATA_STRUCTURES_H_INCLUDED
#define DATA_STRUCTURES_H_INCLUDED
#include <iostream>
using namespace std;

//---------------------- Conclusion List Class--------------------------------------------------

class ConclusionList{

private:
    int rule_number;
    string conclusion_var;
    string varValue;
    bool status;

public:
    ConclusionList(){
        rule_number = 0;
        status=0;
    }

    void set_rule(int ruleNo, string var){
        rule_number = ruleNo;
        conclusion_var = var;
    }

    void set_value(string value)
    {
        varValue = value;
        status = 1;
    }

    void print_rule(){
        cout<< rule_number<<" "<<conclusion_var;
    }

    string get_conclusion()
    {
        return conclusion_var;
    }

    string get_conclusion_value()
    {
        return varValue;
    }

    int get_rule()
    {
        return rule_number;
    }

    void set_status()
    {
        status = 1;
    }

    void set_status(int v)
    {
        status = v;
    }

    bool get_status()
    {
        return status;
    }
};
//---------------------- Variable List Class----------------------------------------------------
class VariableList{

private:
    string varName;
          string varValue;
    bool varStatus;
```

```
public:

        void set(string var, string Value){
                varName = var;
                varValue = Value;
        }

   void set(string Value){
                varValue = Value;
        }
        /*void update(string var, string val){
                varValue = Value;
                //varStatus = "I";
        }*/

        void init(bool stat, string var, string Value){
                varStatus = stat;
                varName = var;
                varValue = Value;
        }

   string getVarName(){
          return varName;
        }
```

```
     string getValue(){
      return varValue;
     }

  void setStatus(bool value)
     {
       varStatus = value;
     }

     bool getStatus()
     {
       return varStatus;
     }

     void printVarList(){
          cout<<varStatus<<"  "<<varName << "  " << varValue;
     }

};

class ClauseVarList
{
private:
  int clause_no;
```

```
    string clause_value;

public:

    void set_vars(int no, string value)
    {
        clause_no = no;
        clause_value = value;
    }

    string get_clause_vars()
    {
        return clause_value;
    }

    void print_clause_vars()
    {
        cout<<"  "<<clause_no<<"  "<<clause_value;
    }

};
#endif // DATA_STRUCTURES_H_INCLUDED
```

6.2 Forward Chaining

forward.cpp

Created an object to *Forward_Chaining* class in *main.cpp*

```
/*** FORWARD CHAINING ***/

#include<iostream>
#include <stdio.h>
#include <string.h>
#include <fstream>
using namespace std;



class Forward_Chaining{

private:

const int BLOCK_SIZE = 2;

int flag;

char cndvar[25][19];
char varlt[26][19],  clvarlt[50][19];
char c[20], vp[19], v[19];
char CANCER_TYPE[30], TREATMENT[50];
int instlt[26];
```

```
#define cndvar_size 25
#define varlt_size 26
#define clvarlt_size 50
#define instlt_size 26

int f, i, j, k, s;
int fp;
int bp;
int sn;
int cn;
int choice;
char answer;

public:
Forward_Chaining(int);
void initialization(void);
void search(void);
void check_instantiation(void);
void instantiate(void);
void cancerListDisplay();
void getTreatment();
 void forward();
            int getChoice(){
            return choice;
            }

            void setChoice(int no){
             choice = no;
            }
            void write_file(string);

};

Forward_Chaining::Forward_Chaining(int no){
choice = no;
forward();
}

 void Forward_Chaining:: forward()
{

    cout << endl;
            cout << endl<<"*******RECOMMENDED TREATMENT FOR DETECTED
CANCER*******\n"<<endl;

            initialization();

            getTreatment();

}

void Forward_Chaining:: initialization(void)
{
    fp=1;
    bp=1;

    for (i=1;i < clvarlt_size; i++)
```

```
        strcpy(clvarlt[i], "");
    for (i=1;i < cndvar_size; i++)
        strcpy(cndvar[i], "");
    for (i=1;i < instlt_size; i++)
        instlt[i]=0;
    for (i=1;i < varlt_size; i++)
        strcpy(varlt[i], "");

    for (i=1;i < cndvar_size; i++)
    {
        strcpy(cndvar[i], "");
        strcpy(varlt[i], "");
        instlt[i]=0;
    }

    strcpy(varlt[1], "CANCER_TYPE");

   for(i=1;i<clvarlt_size+1;i++)
    {
        if(i%2 == 1)
        {
            strcpy(clvarlt[i], "CANCER_TYPE");
        }
    }

        getchar();
}
```

```
void Forward_Chaining:: instantiate()
{
   i=1;

   while ((strcmp(v, varlt[i]) != 0) && (i <= varlt_size))
      i=i+1;

   instlt[i] = 1;
   i = 1;

   while ((strcmp(v, cndvar[i]) != 0) && (i <= cndvar_size))
      i=i+1;

   if (strcmp(v, cndvar[i]) != 0)
   {
      strcpy(cndvar[bp], v);
      bp=bp+1;
   }
}

void Forward_Chaining::getTreatment()
{
   strcpy(c,"CANCER_TYPE");
   strcpy(cndvar[bp], c);
   bp = bp + 1;
   sn = 1; cn = 1;
   f=1;
   Find: search();
   cn=1;
```

```
if (sn != 0)
 {

    i = 2 * (sn-1) + cn;
    strcpy(v, clvarlt[i]);
    while (strcmp(v, "") !=0)
    {
       check_instantiation();
       cn = cn+1;
       i = 2 * (sn-1) + cn;
       strcpy(v, clvarlt[i]);
    }

    s = 0;

    switch(sn)
    {
    case 1:
       if (strcmp(CANCER_TYPE, "large_cell_neuroma") == 0)
          s=1;
       break;
    case 2:
       if ( strcmp(CANCER_TYPE, "squanors_cell_carcinoma")  == 0)
          s=1;
       break;
    case 3:
       if (strcmp(CANCER_TYPE, "large_cell_carcinoma") == 0)
          s=1;
    break;
case 4:
    if ( strcmp(CANCER_TYPE, "adeno_carcinoma") == 0)
        s=1;
    break;

case 5:
    if (strcmp(CANCER_TYPE, "no_lung_cancer") == 0)
        s=1;
    break;
case 6:
    if (strcmp(CANCER_TYPE, "nodular") == 0)
        s=1;
    break;
case 7:
    if (strcmp(CANCER_TYPE, "acral_lentigious") == 0)
        s=1;
    break;
case 8:
    if (strcmp(CANCER_TYPE, "lentigo") == 0)
        s=1;
    break;
case 9:
    if (strcmp(CANCER_TYPE, "superficial_spreading") == 0)
        s=1;
    break;
case 10:
    if ( strcmp(CANCER_TYPE, "no_skin_cancer") == 0)
        s=1;
    break;
case 11:
    if (strcmp(CANCER_TYPE, "leukemia_stage1") == 0)
        s=1;
    break;
case 12:
    if (strcmp(CANCER_TYPE, "acute_myclogeneous") == 0)
        s=1;
    break;
case 13:
    if ( strcmp(CANCER_TYPE, "chronic_myclogeneous") == 0) //Acute Mylogenous
        s=1;
    break;
case 14:
    if (strcmp(CANCER_TYPE, "chronic_lymphocytic") == 0)
        s=1;
    break;
case 15:
    if (strcmp(CANCER_TYPE, "acute_lymphocytic") == 0)
        s=1;
    break;
case 16:
    if (strcmp(CANCER_TYPE, "no_blood_cancer") == 0)
        s=1;
    break;
case 17:
    if (strcmp(CANCER_TYPE, "acoustic_neuroma") == 0)
        s=1;
    break;
```

```
case 18:
    if (strcmp(CANCER_TYPE, "cns_lymphoma") == 0)
        s=1;
    break;
case 19:
    if (strcmp(CANCER_TYPE, "medulloblastoma") == 0)
        s=1;
    break;
case 20:
    if (strcmp(CANCER_TYPE, "pituitary_tumour") == 0)
        s=1;
    break;

case 21:
    if (strcmp(CANCER_TYPE, "enalangio_carcinoma") == 0)
        s=1;
    break;
case 22:
    if (strcmp(CANCER_TYPE, "heptoblastoma") == 0)
        s=1;
    break;
case 23:
    if (strcmp(CANCER_TYPE, "metastasis") == 0)
        s=1;
    break;
case 24:
    if (strcmp(CANCER_TYPE, "angiosarcoma") == 0)
        s=1;
    break;
```

```
 case 25:
    if (strcmp(CANCER_TYPE, "wheezing") == 0)
        s=1;
    break;

}

if (s != 1)
{
    f = sn + 1;
    goto Find;
}

string treat;

switch (sn)
{

case 1:
    {
    strcpy(TREATMENT, "SURGERY");

    cout <<"*      TREATMENT :" << TREATMENT
         << "               *\n";
    cout <<"*************************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
```

```
   instantiate();
   break;
   }

case 2:
   strcpy(TREATMENT, "IMMUNO THERAPY");

   cout <<"*      TREATMENT :" << TREATMENT
       << "               *\n";
   cout <<"************************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;

case 3:
   strcpy(TREATMENT, "TARGETED THERAPY");

   cout <<"*      TREATMENT :" << TREATMENT
       << "               *\n";
   cout <<"************************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;
case 4:
   strcpy(TREATMENT, "CHEMO SURGERY");
   cout <<"*      TREATMENT :" << TREATMENT
       << "              *\n";
   cout <<"***********************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;

//no skin cancer
case 5:
   strcpy(TREATMENT, "PRESCRIBED MEDICATION");

   cout <<"*      TREATMENT :" << TREATMENT
       << "              *\n";
   cout <<"***********************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;

 //skin cancer
case 6:
   strcpy(TREATMENT, "DERMOSCOPY");

   cout <<"*      TREATMENT :" << TREATMENT
       << "              *\n";
```

```
    cout <<"************************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
    break;
case 7:
    strcpy(TREATMENT, "LYMPHANGIOGRAM");

    cout <<"*      TREATMENT :" << TREATMENT
         << "               *\n";
    cout <<"************************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
    break;
case 8:
    strcpy(TREATMENT, "PATHOLOGY");

    cout <<"*      TREATMENT :" << TREATMENT
         << "               *\n";
    cout <<"************************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
    break;
case 9:
    strcpy(TREATMENT, "BIOPSY");

    cout <<"*      TREATMENT :" << TREATMENT
         << "               *\n";
    cout <<"************************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
    break;

   //no skin cancer
case 10:
    strcpy(TREATMENT, "GENERAL_MEDICATION");

    cout <<"*      TREATMENT :" << TREATMENT
         << "               *\n";
    cout <<"************************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
    break;

case 11:
    strcpy(TREATMENT, "GENERAL_MEDICATION");

    cout <<"*      TREATMENT :" << TREATMENT
         << "               *\n";
    cout <<"****************************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
    break;

 //blood cancer
case 12:
    strcpy(TREATMENT, "MONOCLONAL_ANTIBODIES");

    cout <<"*      TREATMENT :" << TREATMENT
         << "              *\n";
    cout <<"****************************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
    break;
case 13:
    strcpy(TREATMENT, "BIOLOGICAL_THERAPY");

    cout <<"*      TREATMENT :" << TREATMENT
         << "              *\n";
    cout <<"****************************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
```

```
   break;
case 14:
   strcpy(TREATMENT, "LEUKAPHORESIS");

   cout <<"*      TREATMENT :" << TREATMENT
       << "              *\n";
   cout <<"*******************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;
case 15:
   strcpy(TREATMENT, "SPECIMEN_BIOPSY");

   cout <<"*      TREATMENT :" << TREATMENT
       << "              *\n";
   cout <<"*******************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;
case 16:
   strcpy(TREATMENT, "GENERAL_MEDICATION");

   cout <<"*      TREATMENT :" << TREATMENT
       << "              *\n";
   cout <<"*******************************************************\n";
```

```
  cout << endl;
  write_file(TREATMENT);
  strcpy(v, "TREATMENT");
  instantiate();
  break;
  case 17:

  strcpy(TREATMENT, "RADIO_SURGERY");

  cout <<"*      TREATMENT :" << TREATMENT
     << "              *\n";
  cout <<"*******************************************************\n";
  cout << endl;
  write_file(TREATMENT);
  strcpy(v, "TREATMENT");
  instantiate();
  break;

case 18:
  strcpy(TREATMENT, "STERIODS");

  cout <<"*      TREATMENT :" << TREATMENT
     << "              *\n";
  cout <<"*******************************************************\n";
  cout << endl;
  write_file(TREATMENT);
  strcpy(v, "TREATMENT");
  instantiate();
  break;
```

```
case 19:
   strcpy(TREATMENT, "STEMCELL_TRANSPLANT");

   cout <<"*      TREATMENT :" << TREATMENT
       << "              *\n";
   cout <<"*****************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;

case 20:
   strcpy(TREATMENT, "TRANSPHENODIAL_SURGERY");

   cout <<"*      TREATMENT :" << TREATMENT
       << "              *\n";
   cout <<"*****************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;

   case 21:
   strcpy(TREATMENT, "PHOTODYNAMIC THERAPY");

   cout <<"*      TREATMENT :" << TREATMENT
```

```
       << "               *\n";
   cout <<"************************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;

case 22:
   strcpy(TREATMENT, "PERCUTANEOUS ETHANOL INJECTION");

   cout <<"*      TREATMENT :" << TREATMENT
       << "               *\n";
   cout <<"************************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
   instantiate();
   break;

case 23:
   strcpy(TREATMENT, "HORMONAL THERAPY");

   cout <<"*      TREATMENT :" << TREATMENT
       << "               *\n";
   cout <<"************************************************************\n";
   cout << endl;
   write_file(TREATMENT);
   strcpy(v, "TREATMENT");
```

```
    instantiate();
    break;

case 24:
    strcpy(TREATMENT, "BIOLOGICAL THERAPY");

    cout <<"*      TREATMENT :" << TREATMENT
         << "              *\n";
    cout <<"*********************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
    break;

 case 25:
    strcpy(TREATMENT, "NEBULIZATION");

    cout <<"*      TREATMENT :" << TREATMENT
         << "              *\n";
    cout <<"*********************************************************\n";
    cout << endl;
    write_file(TREATMENT);
    strcpy(v, "TREATMENT");
    instantiate();
    break;



    }
```

```
            f = sn + 1;
      goto Find;
     }


    fp=fp+1;
    if (fp < bp)
    {
      f = 1;
      goto Find;
    }
}

void Forward_Chaining::check_instantiation(void)
{
         i=1;

   while ((strcmp(v, varlt[i]) != 0) && (i < 2))
         i = i+1;

   if (instlt[i] != 1)
   {

      instlt[i] = 1;
      switch (i)
      {

      case 1:
```

```
{
   repeat:

   switch(getChoice())
   {
       case 1:
          strcpy(CANCER_TYPE, "large_cell_neuroma");
          break;
       case 2:
          strcpy(CANCER_TYPE, "squanors_cell_carcinoma");
          break;
       case 3:
          strcpy(CANCER_TYPE, "large_cell_carcinoma");
          break;
       case 4:
          strcpy(CANCER_TYPE, "adeno_carcinoma");
          break;

       //Skin Cancer
       case 5:
          strcpy(CANCER_TYPE, "no_lung_cancer");
          break;
       case 6:
          strcpy(CANCER_TYPE, "nodular");
          break;
       case 7:
          strcpy(CANCER_TYPE, "acral_lentigious");
          break;
       case 8:
```

```
    strcpy(CANCER_TYPE, "lentigo");
    break;
case 9:
    strcpy(CANCER_TYPE, "superficial_spreading");
    break;
case 10:
    strcpy(CANCER_TYPE, "no_skin_cancer");
    break;
case 11:
    strcpy(CANCER_TYPE, "leukemia_stage1");
    break;
case 12:
    strcpy(CANCER_TYPE, "acute_myclogeneous");
    break;
case 13:
    strcpy(CANCER_TYPE, "chronic_myclogeneous"); //Acute Mylogenous
    break;
case 14:
    strcpy(CANCER_TYPE, "chronic_lymphocytic");
        s=1;
    break;
case 15:
    strcpy(CANCER_TYPE, "acute_lymphocytic");
    break;
case 16:
    strcpy(CANCER_TYPE, "no_blood_cancer");
    break;
case 17:
    strcpy(CANCER_TYPE, "acoustic_neuroma");
```

```
            break;
        case 18:
            strcpy(CANCER_TYPE, "cns_lymphoma");
            break;
        case 19:
            strcpy(CANCER_TYPE, "medulloblastoma");
            break;
        case 20:
            strcpy(CANCER_TYPE, "pituitary_tumour");
            break;

        case 21:strcpy(CANCER_TYPE, "enalangio_carcinoma");
            break;
        case 22:strcpy(CANCER_TYPE, "heptoblastoma");
            break;
        case 23:strcpy(CANCER_TYPE, "metastasis");
            break;
        case 24:strcpy(CANCER_TYPE, "angiosarcoma");
            break;
         case 25:strcpy(CANCER_TYPE, "wheezing");
            break;

        default:
            cout << "\n Invalid choice... "<< endl;
            goto repeat;
       }
            }
   break;
  }
```

```
         }
}


void Forward_Chaining::search()
{
  flag = 0;
   sn = f;

   while ((flag == 0) && (sn <= cndvar_size))
   {
      cn=1;
      k = (sn-1)*BLOCK_SIZE+cn;
      while ((strcmp(clvarlt[k], cndvar[fp]) != 0) && (cn < 3))
      {
         cn = cn+1;
         k = (sn-1)*BLOCK_SIZE+cn;
      }

      if (strcmp(clvarlt[k], cndvar[fp]) == 0)
         flag = 1;

      if (flag == 0)
         sn = sn+1;
   }
   if (flag == 0)
      sn=0;
}
```

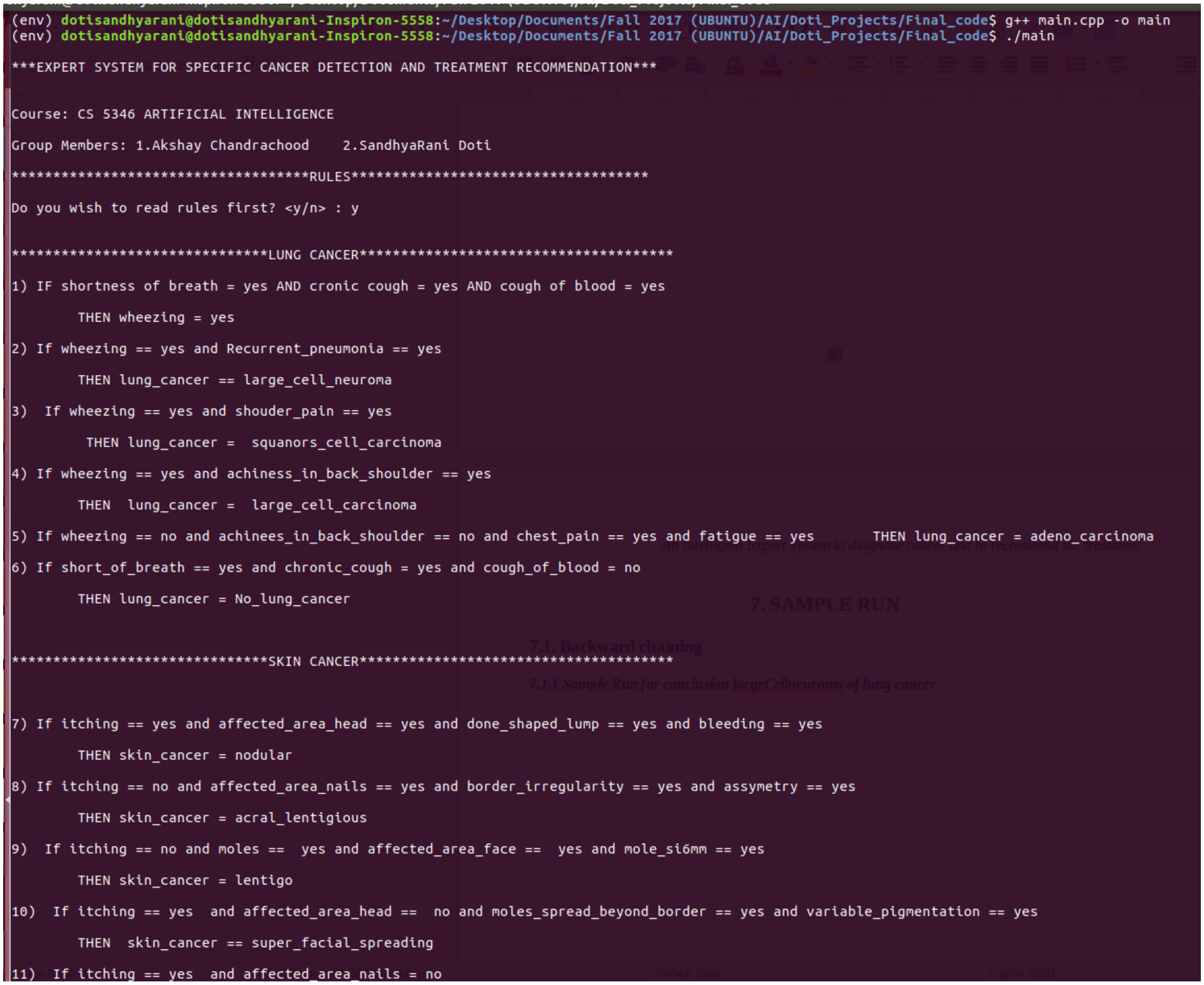
```
void Forward_Chaining::write_file(string treat){
    ofstream fout;
    fout.open("data_log.txt",ios::app);
    fout<<"TREATMENT"<<" :\t"<<treat<<endl;
     fout<<"____________________________________________________________________________________
__________"<<endl<<endl;
}
```

7. SAMPLE RUN

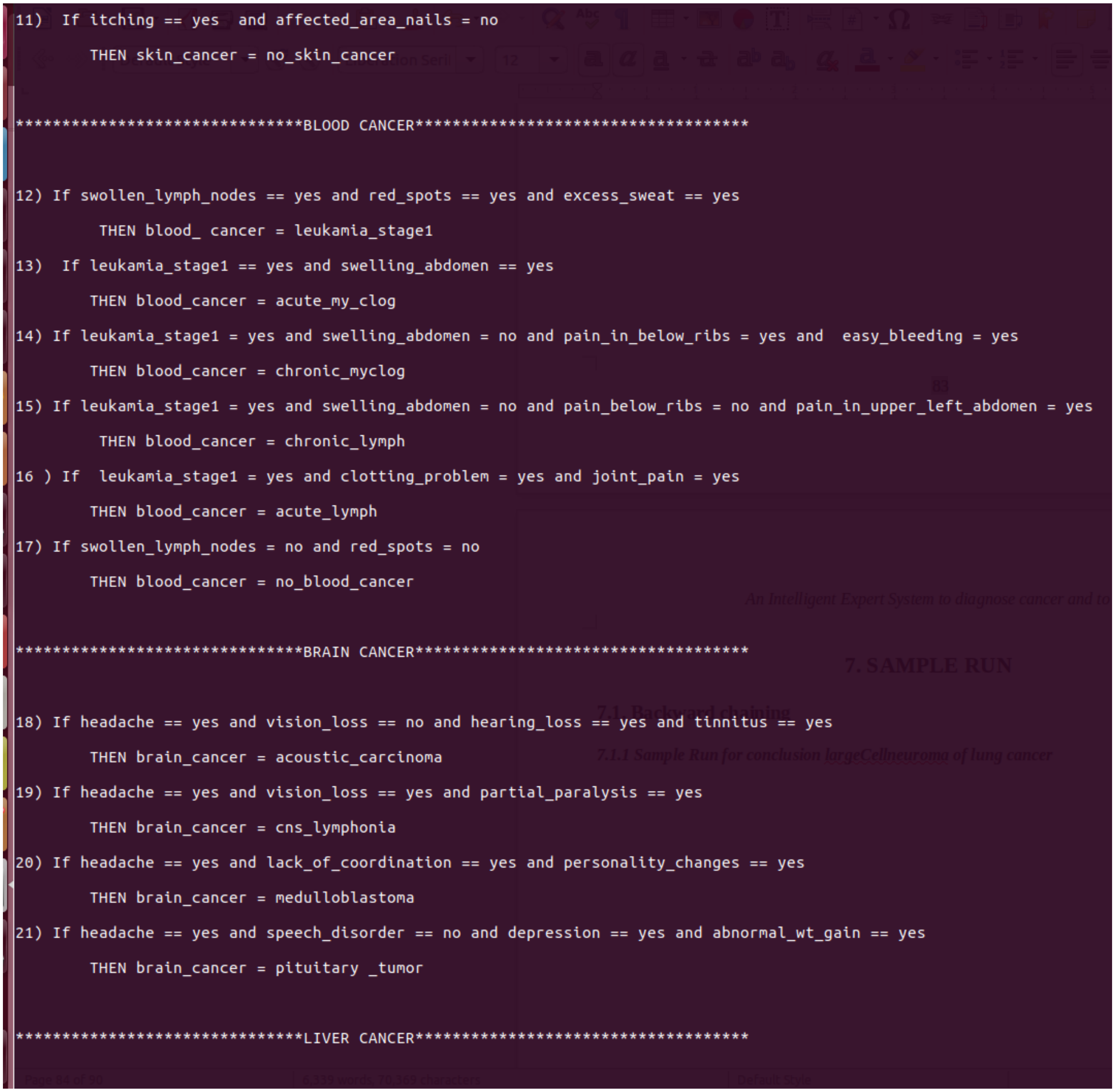
7.1. Backward chaining and Forward Chaining

These are the sample runs for detecting cancer type (backward chaining) and its treatment (forward chaining).

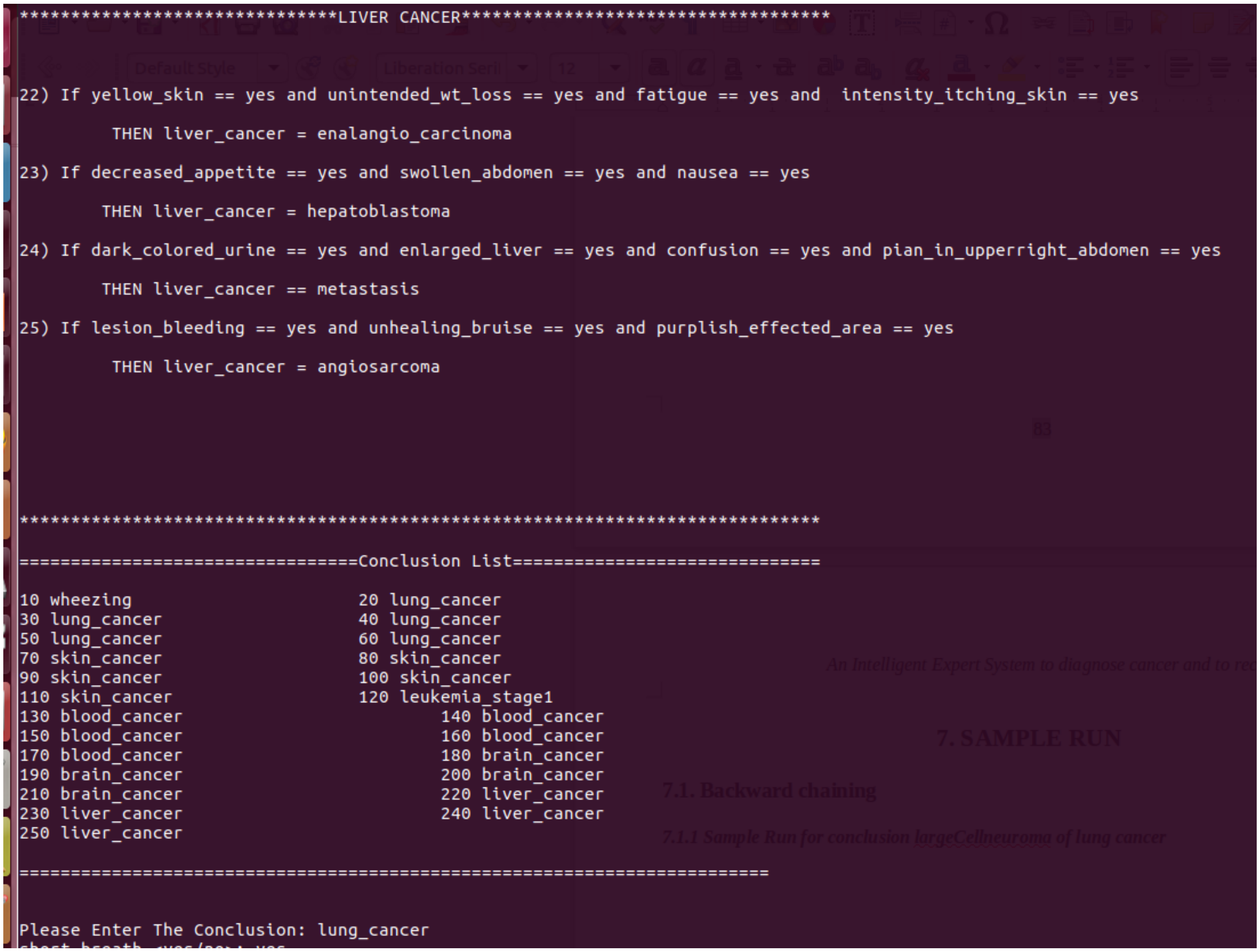
7.1.1 Sample Run for conclusion largeCellneuroma of lung cancer and its treatment



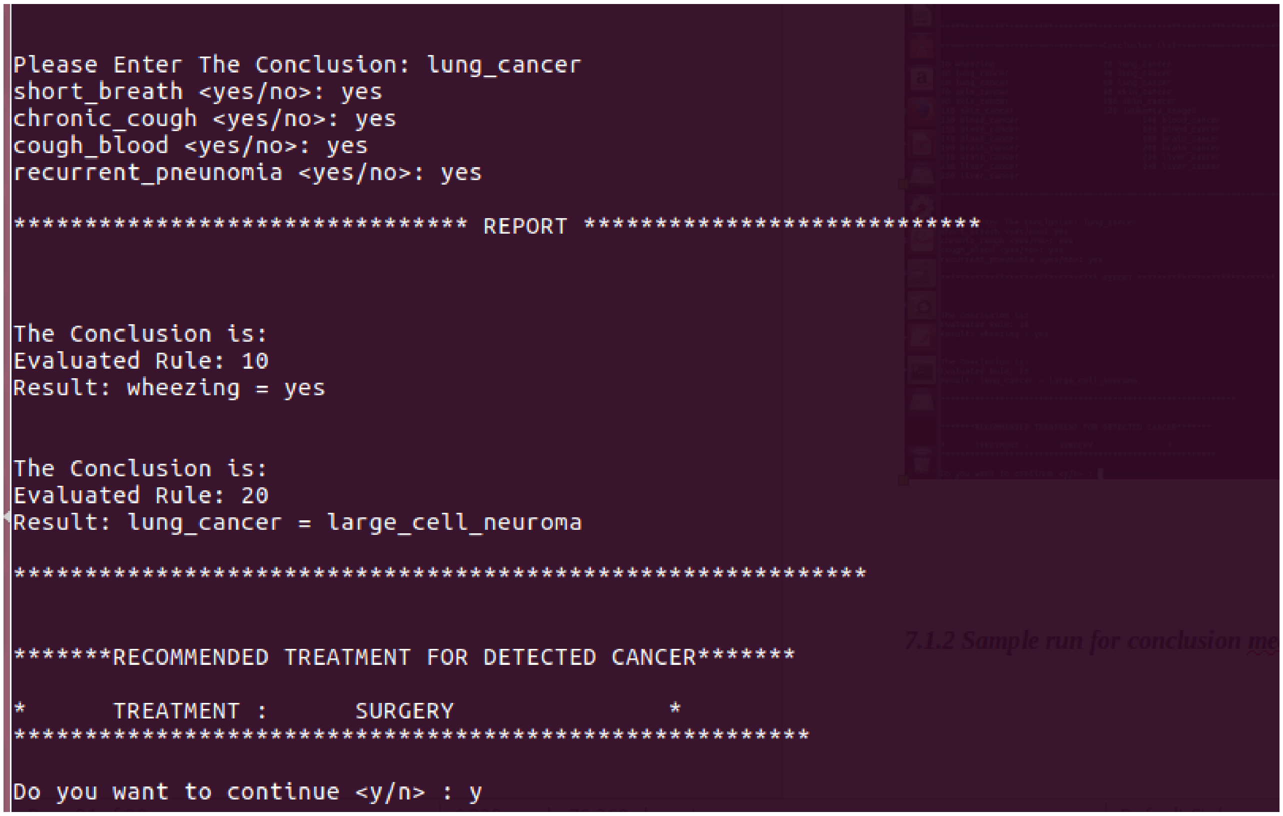
```
(env) dotisandhyarani@dotisandhyarani-Inspiron-5558:~/Desktop/Documents/Fall 2017 (UBUNTU)/AI/Doti_Projects/Final_code$ g++ main.cpp -o main
(env) dotisandhyarani@dotisandhyarani-Inspiron-5558:~/Desktop/Documents/Fall 2017 (UBUNTU)/AI/Doti_Projects/Final_code$ ./main

***EXPERT SYSTEM FOR SPECIFIC CANCER DETECTION AND TREATMENT RECOMMENDATION***

Course: CS 5346 ARTIFICIAL INTELLIGENCE

Group Members: 1.Akshay Chandrachood    2.SandhyaRani Doti

************************************RULES************************************

Do you wish to read rules first? <y/n> : y

*******************************LUNG CANCER**************************************

1) IF shortness of breath = yes AND cronic cough = yes AND cough of blood = yes

        THEN wheezing = yes

2) If wheezing == yes and Recurrent_pneumonia == yes

        THEN lung_cancer == large_cell_neuroma

3)  If wheezing == yes and shouder_pain == yes

         THEN lung_cancer =  squanors_cell_carcinoma

4) If wheezing == yes and achiness_in_back_shoulder == yes

        THEN  lung_cancer =  large_cell_carcinoma

5) If wheezing == no and achinees_in_back_shoulder == no and chest_pain == yes and fatigue == yes       THEN lung_cancer = adeno_carcinoma

6) If short_of_breath == yes and chronic_cough = yes and cough_of_blood = no

        THEN lung_cancer = No_lung_cancer


*******************************SKIN CANCER**************************************


7) If itching == yes and affected_area_head == yes and done_shaped_lump == yes and bleeding == yes

        THEN skin_cancer = nodular

8) If itching == no and affected_area_nails == yes and border_irregularity == yes and assymetry == yes

        THEN skin_cancer = acral_lentigious

9)  If itching == no and moles ==  yes and affected_area_face ==  yes and mole_si6mm == yes

        THEN skin_cancer = lentigo

10)  If itching == yes  and affected_area_head ==  no and moles_spread_beyond_border == yes and variable_pigmentation == yes

        THEN  skin_cancer == super_facial_spreading

11)  If itching == yes  and affected_area_nails = no
```



```
11)  If itching == yes  and affected_area_nails = no

      THEN skin_cancer = no_skin_cancer

*******************************BLOOD CANCER************************************


12) If swollen_lymph_nodes == yes and red_spots == yes and excess_sweat == yes

         THEN blood_ cancer = leukamia_stage1

13)  If leukamia_stage1 == yes and swelling_abdomen == yes

        THEN blood_cancer = acute_my_clog

14) If leukamia_stage1 = yes and swelling_abdomen = no and pain_in_below_ribs = yes and  easy_bleeding = yes

        THEN blood_cancer = chronic_myclog

15) If leukamia_stage1 = yes and swelling_abdomen = no and pain_below_ribs = no and pain_in_upper_left_abdomen = yes

         THEN blood_cancer = chronic_lymph

16 ) If  leukamia_stage1 = yes and clotting_problem = yes and joint_pain = yes

        THEN blood_cancer = acute_lymph

17) If swollen_lymph_nodes = no and red_spots = no

        THEN blood_cancer = no_blood_cancer


*******************************BRAIN CANCER************************************


18) If headache == yes and vision_loss == no and hearing_loss == yes and tinnitus == yes

        THEN brain_cancer = acoustic_carcinoma

19) If headache == yes and vision_loss == yes and partial_paralysis == yes

        THEN brain_cancer = cns_lymphonia

20) If headache == yes and lack_of_coordination == yes and personality_changes == yes

        THEN brain_cancer = medulloblastoma

21) If headache == yes and speech_disorder == no and depression == yes and abnormal_wt_gain == yes

        THEN brain_cancer = pituitary _tumor


*******************************LIVER CANCER************************************
```

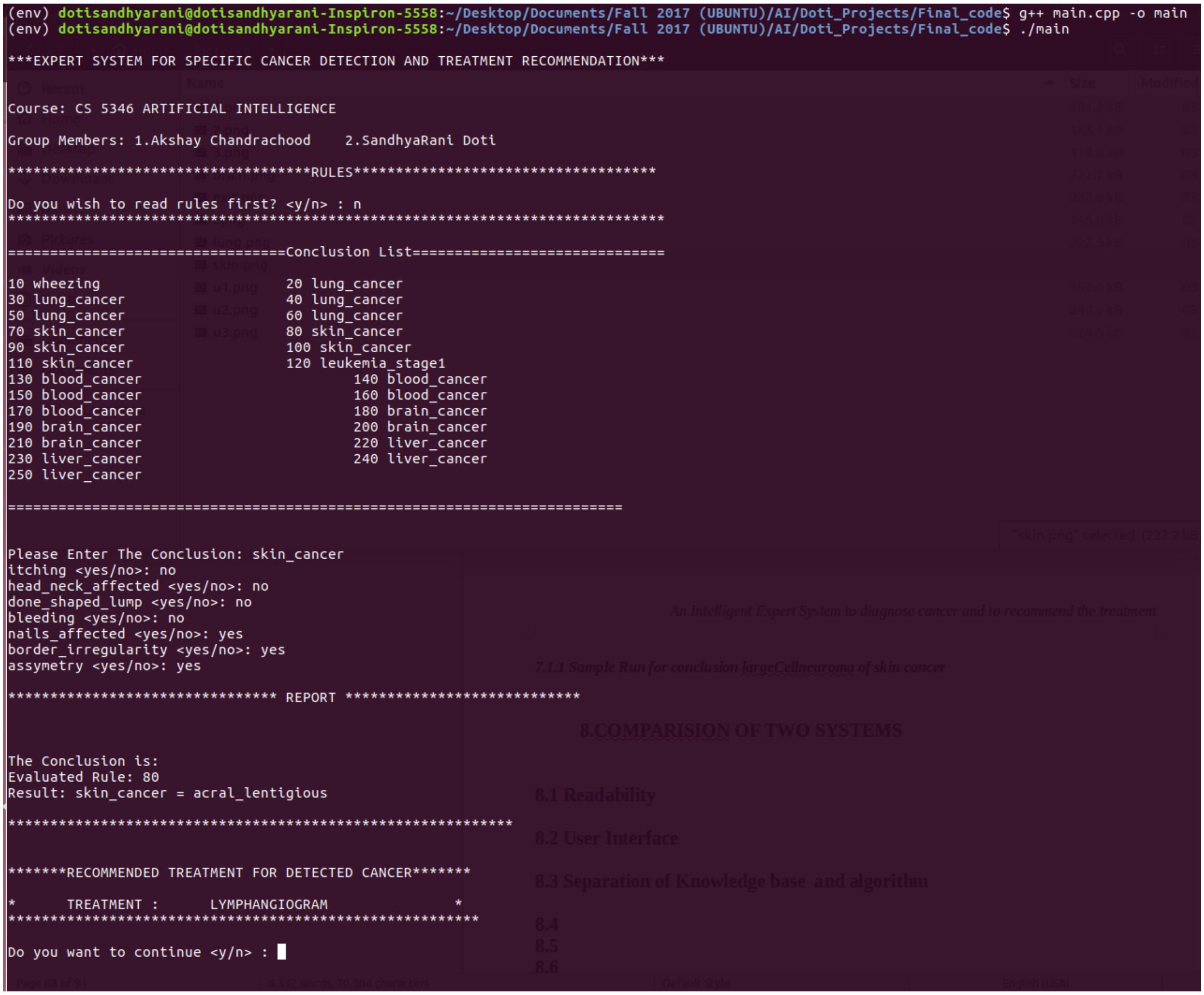


```
*******************************LIVER CANCER************************************

22) If yellow_skin == yes and unintended_wt_loss == yes and fatigue == yes and  intensity_itching_skin == yes

         THEN liver_cancer = enalangio_carcinoma

23) If decreased_appetite == yes and swollen_abdomen == yes and nausea == yes

        THEN liver_cancer = hepatoblastoma

24) If dark_colored_urine == yes and enlarged_liver == yes and confusion == yes and pian_in_upperright_abdomen == yes

        THEN liver_cancer == metastasis

25) If lesion_bleeding == yes and unhealing_bruise == yes and purplish_effected_area == yes

         THEN liver_cancer = angiosarcoma




*******************************************************************************

=================================Conclusion List==============================

10 wheezing                        20 lung_cancer
30 lung_cancer                     40 lung_cancer
50 lung_cancer                     60 lung_cancer
70 skin_cancer                     80 skin_cancer
90 skin_cancer                     100 skin_cancer
110 skin_cancer                    120 leukemia_stage1
130 blood_cancer                           140 blood_cancer
150 blood_cancer                           160 blood_cancer
170 blood_cancer                           180 brain_cancer
190 brain_cancer                           200 brain_cancer
210 brain_cancer                           220 liver_cancer
230 liver_cancer                           240 liver_cancer
250 liver_cancer

==========================================================================


Please Enter The Conclusion: lung_cancer
short_breath <yes/no>: yes
```



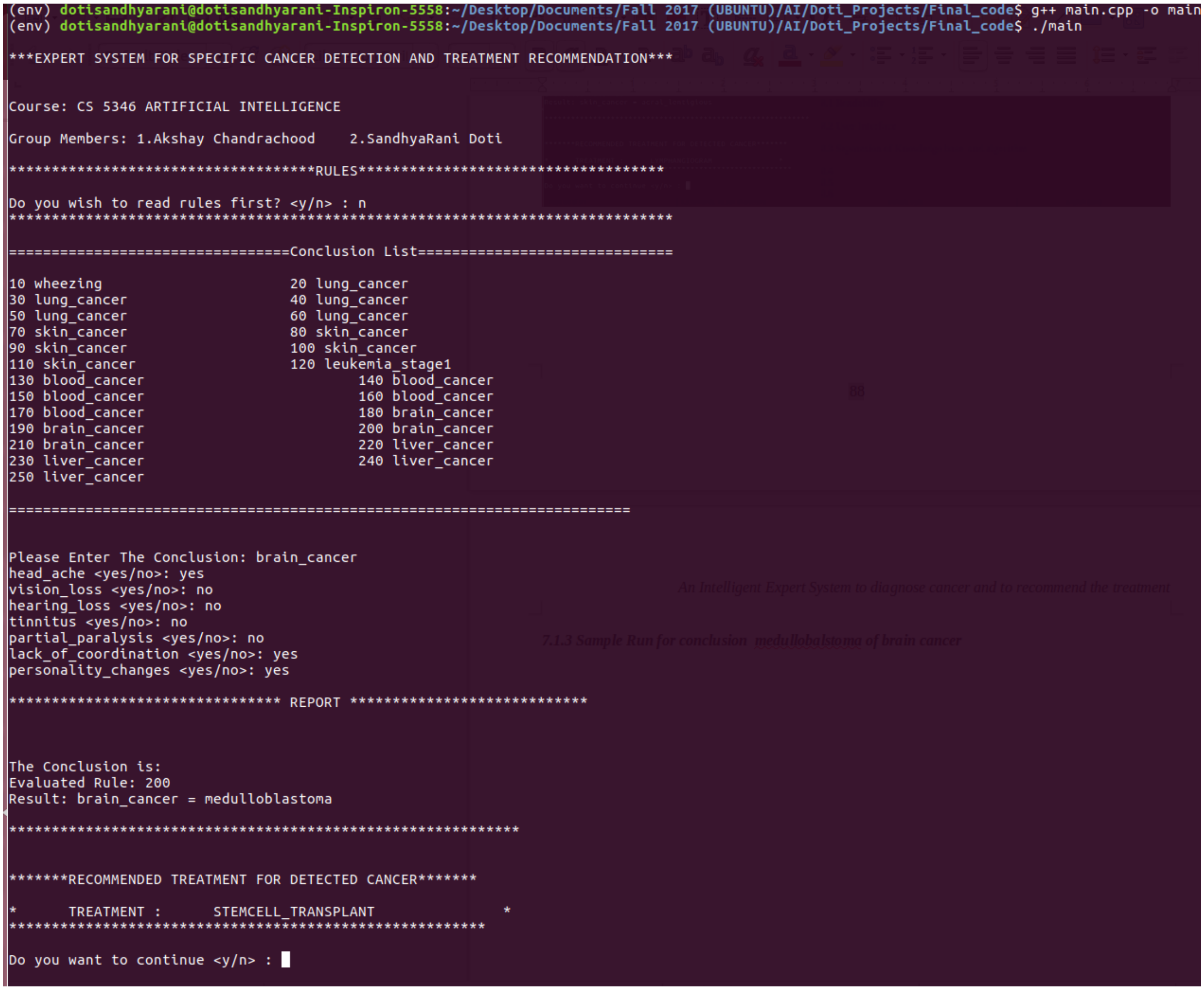
```
Please Enter The Conclusion: lung_cancer
short_breath <yes/no>: yes
chronic_cough <yes/no>: yes
cough_blood <yes/no>: yes
recurrent_pneunomia <yes/no>: yes

******************************** REPORT ****************************



The Conclusion is:
Evaluated Rule: 10
Result: wheezing = yes


The Conclusion is:
Evaluated Rule: 20
Result: lung_cancer = large_cell_neuroma

************************************************************


*******RECOMMENDED TREATMENT FOR DETECTED CANCER*******

*      TREATMENT :      SURGERY               *
********************************************************

Do you want to continue <y/n> : y
```

7.1.2 Sample Run for conclusion acral_lentigious of skin cancer and its treatment



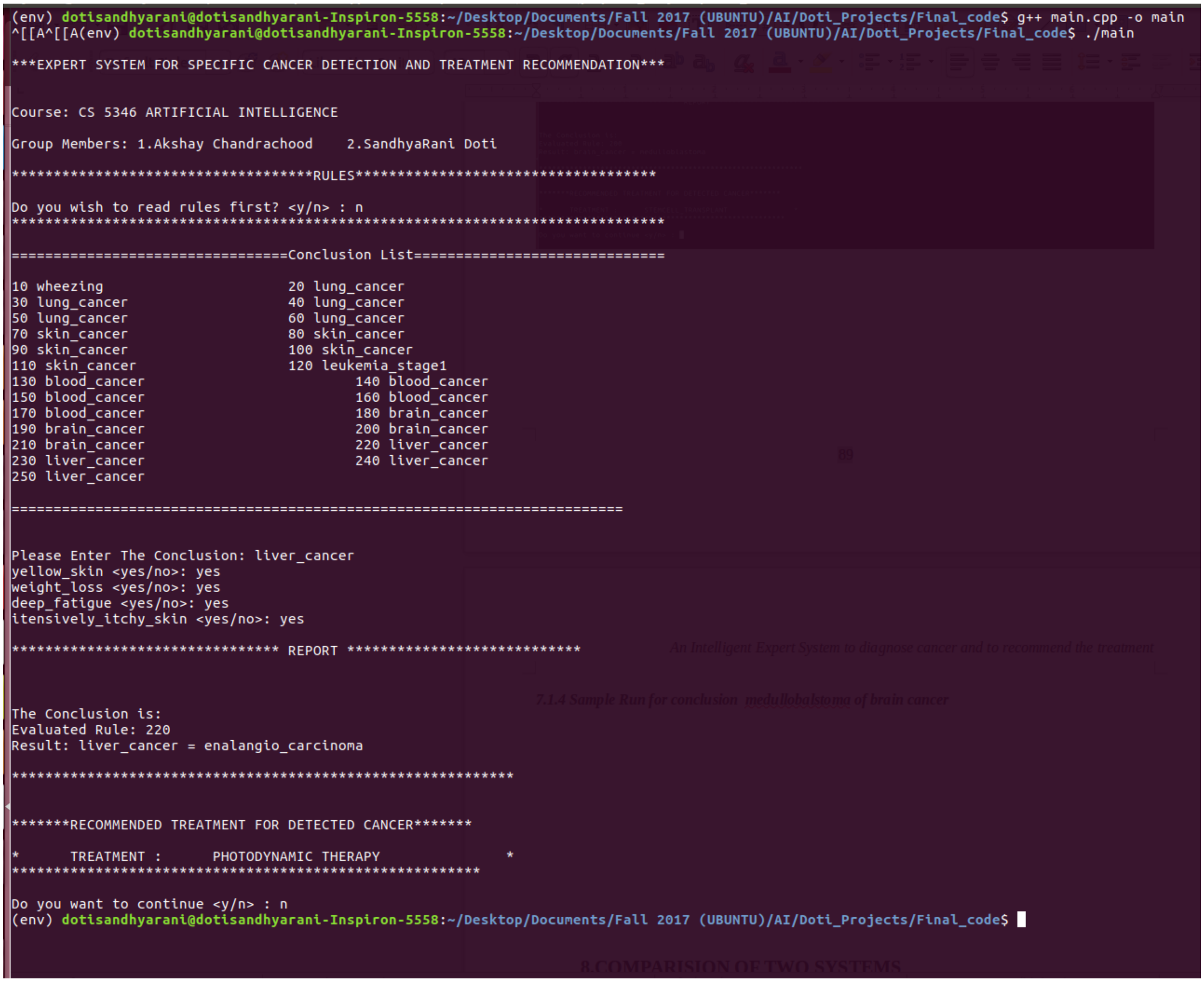
```
(env) dotisandhyarani@dotisandhyarani-Inspiron-5558:~/Desktop/Documents/Fall 2017 (UBUNTU)/AI/Doti_Projects/Final_code$ g++ main.cpp -o main
(env) dotisandhyarani@dotisandhyarani-Inspiron-5558:~/Desktop/Documents/Fall 2017 (UBUNTU)/AI/Doti_Projects/Final_code$ ./main

***EXPERT SYSTEM FOR SPECIFIC CANCER DETECTION AND TREATMENT RECOMMENDATION***

Course: CS 5346 ARTIFICIAL INTELLIGENCE

Group Members: 1.Akshay Chandrachood    2.SandhyaRani Doti

************************************RULES***********************************

Do you wish to read rules first? <y/n> : n
******************************************************************************
=================================Conclusion List==============================

10 wheezing                      20 lung_cancer
30 lung_cancer                   40 lung_cancer
50 lung_cancer                   60 lung_cancer
70 skin_cancer                   80 skin_cancer
90 skin_cancer                   100 skin_cancer
110 skin_cancer                  120 leukemia_stage1
130 blood_cancer                         140 blood_cancer
150 blood_cancer                         160 blood_cancer
170 blood_cancer                         180 brain_cancer
190 brain_cancer                         200 brain_cancer
210 brain_cancer                         220 liver_cancer
230 liver_cancer                         240 liver_cancer
250 liver_cancer

=========================================================================


Please Enter The Conclusion: skin_cancer
itching <yes/no>: no
head_neck_affected <yes/no>: no
done_shaped_lump <yes/no>: no
bleeding <yes/no>: no
nails_affected <yes/no>: yes
border_irregularity <yes/no>: yes
assymetry <yes/no>: yes

******************************** REPORT ****************************


The Conclusion is:
Evaluated Rule: 80
Result: skin_cancer = acral_lentigious

************************************************************


*******RECOMMENDED TREATMENT FOR DETECTED CANCER*******

*      TREATMENT :      LYMPHANGIOGRAM               *
********************************************************

Do you want to continue <y/n> :
```

7.1.3 Sample Run for conclusion medullobalstoma of brain cancer and its treatment



```
(env) dotisandhyarani@dotisandhyarani-Inspiron-5558:~/Desktop/Documents/Fall 2017 (UBUNTU)/AI/Doti_Projects/Final_code$ g++ main.cpp -o main
(env) dotisandhyarani@dotisandhyarani-Inspiron-5558:~/Desktop/Documents/Fall 2017 (UBUNTU)/AI/Doti_Projects/Final_code$ ./main

***EXPERT SYSTEM FOR SPECIFIC CANCER DETECTION AND TREATMENT RECOMMENDATION***

Course: CS 5346 ARTIFICIAL INTELLIGENCE

Group Members: 1.Akshay Chandrachood    2.SandhyaRani Doti

************************************RULES************************************

Do you wish to read rules first? <y/n> : n
******************************************************************************

=================================Conclusion List==============================

10 wheezing                      20 lung_cancer
30 lung_cancer                   40 lung_cancer
50 lung_cancer                   60 lung_cancer
70 skin_cancer                   80 skin_cancer
90 skin_cancer                   100 skin_cancer
110 skin_cancer                  120 leukemia_stage1
130 blood_cancer                         140 blood_cancer
150 blood_cancer                         160 blood_cancer
170 blood_cancer                         180 brain_cancer
190 brain_cancer                         200 brain_cancer
210 brain_cancer                         220 liver_cancer
230 liver_cancer                         240 liver_cancer
250 liver_cancer

=========================================================================


Please Enter The Conclusion: brain_cancer
head_ache <yes/no>: yes
vision_loss <yes/no>: no
hearing_loss <yes/no>: no
tinnitus <yes/no>: no
partial_paralysis <yes/no>: no
lack_of_coordination <yes/no>: yes
personality_changes <yes/no>: yes

******************************** REPORT ****************************



The Conclusion is:
Evaluated Rule: 200
Result: brain_cancer = medulloblastoma

************************************************************


*******RECOMMENDED TREATMENT FOR DETECTED CANCER*******

*      TREATMENT :      STEMCELL_TRANSPLANT              *
********************************************************

Do you want to continue <y/n> : █
```

7.1.4 Sample Run for conclusion enalangio of liver cancer and its treatment



```
(env) dotisandhyarani@dotisandhyarani-Inspiron-5558:~/Desktop/Documents/Fall 2017 (UBUNTU)/AI/Doti_Projects/Final_code$ g++ main.cpp -o main
^[[A^[[A(env) dotisandhyarani@dotisandhyarani-Inspiron-5558:~/Desktop/Documents/Fall 2017 (UBUNTU)/AI/Doti_Projects/Final_code$ ./main

***EXPERT SYSTEM FOR SPECIFIC CANCER DETECTION AND TREATMENT RECOMMENDATION***


Course: CS 5346 ARTIFICIAL INTELLIGENCE

Group Members: 1.Akshay Chandrachood    2.SandhyaRani Doti

************************************RULES************************************

Do you wish to read rules first? <y/n> : n
******************************************************************************

=================================Conclusion List==============================

10 wheezing                      20 lung_cancer
30 lung_cancer                   40 lung_cancer
50 lung_cancer                   60 lung_cancer
70 skin_cancer                   80 skin_cancer
90 skin_cancer                   100 skin_cancer
110 skin_cancer                  120 leukemia_stage1
130 blood_cancer                         140 blood_cancer
150 blood_cancer                         160 blood_cancer
170 blood_cancer                         180 brain_cancer
190 brain_cancer                         200 brain_cancer
210 brain_cancer                         220 liver_cancer
230 liver_cancer                         240 liver_cancer
250 liver_cancer

=========================================================================


Please Enter The Conclusion: liver_cancer
yellow_skin <yes/no>: yes
weight_loss <yes/no>: yes
deep_fatigue <yes/no>: yes
itensively_itchy_skin <yes/no>: yes

******************************** REPORT ****************************


The Conclusion is:
Evaluated Rule: 220
Result: liver_cancer = enalangio_carcinoma

************************************************************


*******RECOMMENDED TREATMENT FOR DETECTED CANCER*******

*      TREATMENT :      PHOTODYNAMIC THERAPY               *
********************************************************

Do you want to continue <y/n> : n
(env) dotisandhyarani@dotisandhyarani-Inspiron-5558:~/Desktop/Documents/Fall 2017 (UBUNTU)/AI/Doti_Projects/Final_code$
```

8. COMPARISON OF TWO SYSTEMS

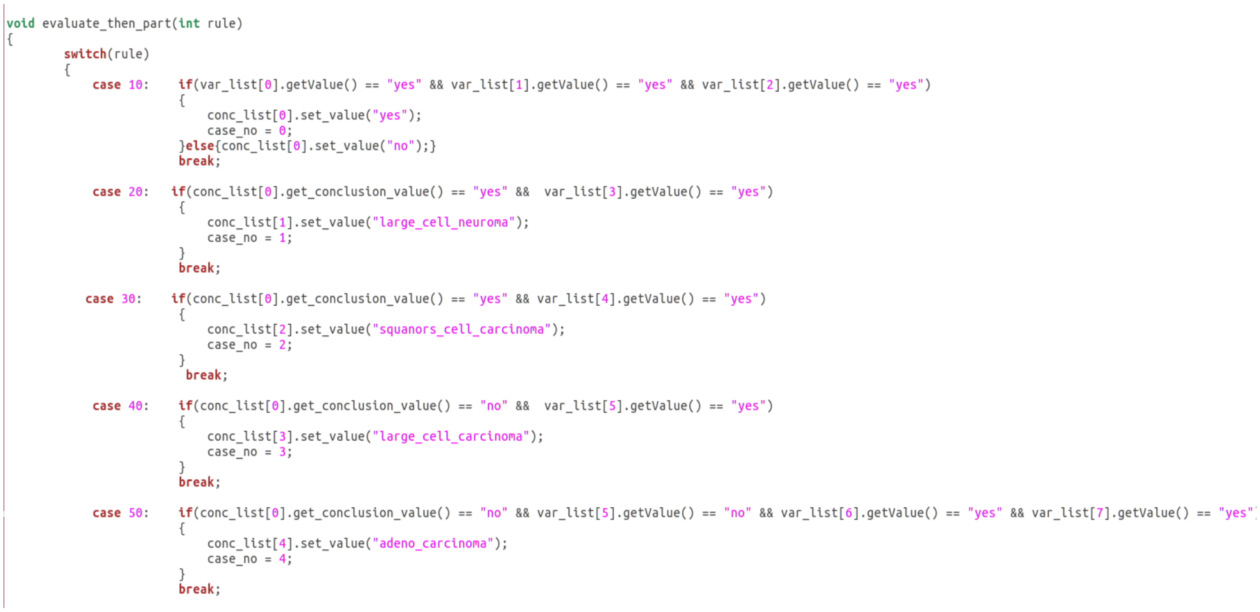
This section will contrast of the system provided by the professor, and the system we developed that is described in this report.

8.1 User Interface

The system displays all types of cancer to the user which is more convenient and clear to the user.

8.2 Separation of Knowledge base and algorithm

The rule statements are stored in a separate function called *evaluate_then_part*.



```
void evaluate_then_part(int rule)
{
        switch(rule)
        {
            case 10:    if(var_list[0].getValue() == "yes" && var_list[1].getValue() == "yes" && var_list[2].getValue() == "yes")
                        {
                            conc_list[0].set_value("yes");
                            case_no = 0;
                        }else{conc_list[0].set_value("no");}
                        break;

            case 20:   if(conc_list[0].get_conclusion_value() == "yes" &&  var_list[3].getValue() == "yes")
                        {
                            conc_list[1].set_value("large_cell_neuroma");
                            case_no = 1;
                        }
                        break;

           case 30:    if(conc_list[0].get_conclusion_value() == "yes" && var_list[4].getValue() == "yes")
                        {
                            conc_list[2].set_value("squanors_cell_carcinoma");
                            case_no = 2;
                        }
                         break;

            case 40:    if(conc_list[0].get_conclusion_value() == "no" &&  var_list[5].getValue() == "yes")
                        {
                            conc_list[3].set_value("large_cell_carcinoma");
                            case_no = 3;
                        }
                        break;

            case 50:    if(conc_list[0].get_conclusion_value() == "no" && var_list[5].getValue() == "no" && var_list[6].getValue() == "yes" && var_list[7].getValue() == "yes"
                        {
                            conc_list[4].set_value("adeno_carcinoma");
                            case_no = 4;
                        }
                        break;
```

9. CONCLUSION

It is a very interesting project, I learned ho w to create a decision tree from the data about the system and generating rules from decision tree and moreover, how to develop an expert system using the inference engines, such as forward chaining and backward chaining algorithms. I gained a good experience when working with many data structures (clause variable list, variable list, conclusion queue) and its flow. I believe an expert system is more accurate to take decision and reduces inconsistency.

10. REFERENCES

- **https://en.wikipedia.org/wiki/Forward_chaining**
- **https://en.wikipedia.org/wiki/Backward_chaining**
- **http://www.cancercenter.com/lung-cancer/symptoms/**
- **http://www.mayoclinic.org/diseases-conditions**
- **Levine, R. AI and Expert Systems: A Comprehensive Guide.**

11. APPENDIX

11.1 Project instruction

Create an intelligent computer expert system for a hospital to diagnose Cancer and to recommend the treatment based on the diagnosis. Perform research using Web or any other source to collect knowledge about the symptoms of Cancer as well as treatments. The hospital staff will feed the symptoms of the patient. Your expert system will diagnose the specific Cancer and will recommend the treatment.

After collecting knowledge, develop two decision tree; one for diagnoses and the other for treatment. Then transform the decision trees into rules. The diagnoses decision tree should be big enough to generate a minimum of twenty five rules. The treatment decision tree should be big enough to cover all types of cancers. The rules should contain variables.

Implement the expert system program, employing Backward Chaining and Forward Chaining methodologies. Programs based on these methodologies are provided on TRACS.

These programs, written in C, are intentionally written poorly and are inefficient and erroneous. Rewrite these programs in C++ by employing Software Engineering principles. Separate Knowledge base and Inference Engine parts of each program and bring efficiency in functionality and output using your creativity. Though you can totally rewrite the programs, they must be based on the methodologies used in these programs. Using any programs from any other source including web will be treated as plagiarism subject to severe punishment.

Develop a user-friendly interface, which receives input data from a clinic staff in restricted English format, uses keyword matching, and responds in a restricted English format.