Texas State University

Artificial Intelligence

Code: CS 5346

Project #1

Expert Cancer Diagnosis

AN INTELLIGENT COMPUTER EXPERT SYSTEM FOR
DIAGNOSING VARIOUS TYPES OF CANCERS

Faculty: Dr. Moonis Ali

Submitted by
Varun Dalal
A04743047

TABLE OF CONTENTS

1. INTRODUCTION:

The objective of this project was to create an intelligent Expert computer system which can detect the type of cancer and then provide the appropriate treatment for it.

2. PROBLEM STATEMENT:

Computer systems that are designed with decision-making capability like a human expert are called as Expert Systems. These Expert systems do not follow any procedures like in the case of conventional programming but are designed to solve complex problems by reasoning about knowledge, like an expert. The requirement of this project is to create an intelligent computer expert system for Cancer Diagnosis.
Although there exist an number of Cancers in the world , the aim of this project was to create a system which includes a knowledgebase which can be easily updated to accommodate various kinds of cancers and therefore only five types of cancers have been included. After diagnosing the diseases, the system should recommend the treatment based on the diagnosis.

3. END USER:

The end User in this system are aimed to be the staff of the diagnosis center which runs the system. The staff is required to enter inputs based on suggestion by the system to get the diagnosis and treatment.

4. PROBLEM DEFINITION:

The problem is to create an intelligent computer expert system for a hospital to diagnose Cancer and to recommend the treatment based on the diagnosis. The hospital staff will feed the symptoms of the patient. The expert system will diagnose the specific Cancer and will recommend the treatment.

4.1 TEAM MEMBERS:

1. **Sidhant Chadha**

2. **Varun Dalal**

4.2 CONTRIBUTIONS TO THE PROJECT:

Varun Dalal:
I worked on collecting symptoms for forward chaining, forward chaining decision tree, converting tree to if then rules and backward chaining coding and forward chaining coding, and collecting treatment information for cancers.

Sidhant Chadha:
Sidhant worked on collecting symptoms for backward chaining, backward chaining decision tree , converting the tree to rules, Backward chaining algorithm and coding, testing the code, and collecting treatment information for cancers..

5. ARTIFICIAL INTELLIGENCE:

The branch of computer science that deals with simulation of intelligent behavior in computers, making those to behave like humans is called Artificial Intelligence. Artificial intelligence includes the following

Games playing: These are the programs that enables the computers to play games like tic-tac-toe or chess.

Expert systems: **These** are the programs that helps doctors to diagnose various diseases and provide treatment by decision making.
Natural language : These programs deals with understanding the natural human language.
Neural networks : Systems **that** simulate intelligence by attempting to reproduce the types of physical connections that occur in animal brains
Robotics : These are the programs that enable the computers to behave like humans including hearing and understanding.

6. RESEARCH IN ARTIFICIAL INTELLIGENCE:

Reasoning
Knowledge
Planning
Learning
Communication
Perception and the ability to move and manipulate objects.

The approaches in Artificial Intelligence includes Traditional symbolic AI, Statistical methods and Computational Intelligence

7. COMPONENTS OF ARTIFICIAL INTELLIGENCE:

7.1 LEARNING:

The ability of computers to find patterns from a sequence of inputs in called as Machine Learning. It includes two types namely Classification and Regression. Classification determines the classification of a program and going through a series of examples. Regression goes through all the inputs and determines a function from that which in turn predicts the output. Computational Learning Theory is a branch of Computer Science which mathematically analyzes the performance of Machine Learning.

7.2 ? REASONING:

In early stages of Artificial Intelligence, computers are developed to think in a step by step process like humans to solve complex problems. But later, it went through a massive development which enabled the

computers to solve even uncertain and incomplete problems. But there are few problems which existed while solving these difficult problems. They include

Requires high computational resources.
Astronomical memory usage.

Even though computers are able to solve problems using step by step process, they are not as quick and reliable as the human being does using their intuitive judgments and consciousness. So, with an idea of achieving human-like thinking ability, Artificial Intelligence made some progress its research into embodies agent, sensorimotor skills and neural networks that increased the probability nature of human-like ability to guess.

7.3 NATURAL LANGUAGE PROCESSING:

Natural Language Processing is the method of understanding the natural languages of humans. It deals with reading and understanding the languages which humans write and speak respectively. This is one of the major area which is more concentrated for development. A good natural language processing system will enable the acquisition of knowledge directly from internet text and extracting its meaning.

Semantic Indexing is the method which is set in processing and retrieving meaning from human languages.
I has few advantages as follows.

Increase in processing speed.
Reduced cost of data storage.
High efficiency in indexing large volumes of abstractions of user input.

7.4 KNOWLEDGE REPRESENTATION:

Knowledge is represented as Ontology, which is a set of concepts within a domain and the relationships between those concepts. Following are the things that are to be represented by Artificial Intelligence.

Objects
Properties
Categories
relations between objects
situations
events
states and time
causes and effects
knowledge about knowledge

8. APPLICATIONS OF ARTIFICIAL INTELLIGENCE:

Hospitals and medicine:

Artificial Intelligence is used in Hospitals and clinics to detect diseases of the patients and provide treatment for them. It is also used to provide medical information of the patients, organize staff rotation and bed schedules.

Finance:

Artificial neural networks are used in banks to manage stocks, properties and organize operations.
Financial Institutions also use Artificial Intelligence to prevent and detect charges out of norms.

Online and telephone customer service:

Artificial Intelligence is used in online telephone and customer service for answering calls from the customers. It used natural language processing for understanding the user queries and responds them back. It is used as the first level of customer support in many companies. Automated answering machines are also implemented in call centers which uses speech recognition, text mining and natural language processing to handle customers. It has a main advantage of reducing the cost of training and operations in enterprises.

Heavy Industry:

In most of the heavy industries, Robots are employed. They are given jobs which may be dangerous to humans and they have also have proved to be more effective than humans in many ways.

Telecommunications:

Heuristic search is used in many telecommunication companies to manage their work forces and scheduling for engineers.

Transportation:

Fuzzy Logic controllers are used in automobiles for automatic gear boxes.

Music:

In the field of Music, Artificial Intelligence is used to compose music like an experienced musician.
The research areas include composition, performance, music theory and sound processing.

Toys and Games:

Artificial Intelligence is used in the field of toys to produce toys such as dogs with some basic Artificial Intelligence suitable for kids. It can also be used in games to act according to the inputs given by the user.

News publishing and writing:

Artificial Intelligence is used to generate news and commercial publications based on statistical data. I also creates real estate analyses and financial reports. It is also used to convert structured data into comments using natural language processing.

9. EXPERT SYSTEMS:

An expert system is a computer program which behaves like a human to provide judgment like an expert and experienced in a field. It mainly consists of two parts namely,

Knowledge base
Inference Engine

9.1 KNOWLEDGE BASE:

In an expert system, IF-Then rules are used to express the knowledge base. Every rule has an IF part, called the antecedent and a THEN part, called the consequent part. When a problem occurs, these rules are used as an evidence to derive a conclusion.

9.2 INFERENCE ENGINE:

Inference engine programs are designed to produce reasoning based on the rules and logics. Inference Engine is developed in two ways,

Forward Chaining
Backward Chaining

ADVANTAGES OF INFERENCE ENGINE:

Expert systems can be written faster than a conventional program by users or experts.
It is able to exploit a considerable amount of knowledge.
It is more reliable and consistent.
It is easy to identify the rules that have to be removed or modified.
The user will be having the information about their problem before the expert systems actually produces the result.

DISADVANTAGES OF INFERENCE ENGINE:

Knowledge collection and its interpretation into rules is a tedious process.
Existence of contradictory rules.
Expert systems may be penalized by the logic used.
Propositional logics uses only invariant facts.
Using variable facts makes the expert system to look less easy and less understandable by the users.

10. APPLICATION OF EXPERT SYSTEM IN HEALTH CARE:

In healthcare, maintaining medical records always involves problems like human errors, lack of space etc. To overcome these defects we use an expert system called Electronic Health Record (EHR), which brings together a more versatile, expansive and robust environment to provide greater quality care. An integrated expert system is set across the hospital by replacing paper based medical records with EHR.

11. INFORMATION GATHERING:

We did a lot of research in learning about various mental disorders, their symptoms and treatment for those disorders. The data which we collected through our research is used to develop the expert system. Our professor Dr.Moonis Ali – Texas State University provided the basic required information of the project.
Based on this information, we browsed a lot of websites and finally were able to collect all the information.

12. DECISION TREES:

Decision trees are used to solve a classification problem whose solution can be predetermined by a set of possible answers. It consist of a parent node or root node from which other nodes get expanded. The nodes that are derived from other node are called as child nodes of that particular parent node. The node which doesn't have children are called leaf nodes. Leaf nodes represent all possible solutions and other nodes are called decision nodes.

Traversing a tree to get an answer is done by beginning at the root node and moving left or right depending upon the decision rule. If the result node is reached, the search is stopped and the current location value is derived from the decision tree.

A tree should be represented as facts instead of rules so that they can be easily modified, added or deleted. Each node of a tree should be represented by fact. The information from one node is stored in a file when it is transferred to another node. Rules for traversal of the tree should be determined enabling the traversal process.

13. RULES FORMATION:

Rules are formed based upon the results of the decision tree. Each and every decision node in a tree consists of a rule which is made of IF and THEN clauses. Rules are formed for both forward and backward chaining.

14. FORMATION OF IF-THEN CLAUSES:

In an IF THEN clause there are two parts namely IF part and CLAUSE part. IF part consists of the condition variables leading to the conclusion node and THEN part consists of the result or conclusion. IF part and THEN parts are connected using any of the logical operators namely AND, OR and NOT. There can be more than one IF clause in a rule but it should not have more than one THEN clause. THEN part of a rule is executed one and only if all the IF parts are true in a rule.

15. ALGORITHMS IMPLEMENTATION:

The final rules are converted into coding using Forward and Backward Chaining Algorithms.

15.1 FORWARD CHAINING (FACT DRIVEN) ALGORITHM AND IMPLEMENTATION:

Forward chaining starts with the available information and uses the rules to extract more data or information by asking questions from the end user until a goal is reached. Inference engine using forward chaining searches the inference rules until it finds a rule in which an IF clause is known to be true. After reaching such a rule, the inference engine can infer the THEN clause and add new information to its data.

15.1.1 ? PROCESSING THE KNOWLEDGE BASE:

The knowledge base of forward chaining begins with creating variable name table and framing of rules based on those variable names.

15.1.2 DATA STRUCTURES USED IN FORWARD CHAINING:

Clause Variable List:

Clause variable list stores all the variables used in IF part of the rules. These variables are stored in array with four array slots allocated for each slot. If only one or two array slots are filled, the remaining slots are left blank.

Conclusion Variable Queue:

It consists of currently executing variables of an IF clause. These variables are stored in queues. If a variable has completed processing, it is removed from the queue and the next variable that has to be processed is added into the queue. When all the variables are processed and there are no more variables, it means the forward chaining is completed and ready to provide answers.

Clause Variable Pointer:

A clause variable pointer keeps track of the current rule and currently executed clause in that rule. It consists of rule number and clause number.

Variable List:

Variable list consists of all variables used in the rules. All the IF clause variables are stored in it and if a variable exists more than one time, it is written only once. Initially no variable will be Instantiated but when a user inputs a value, the variable corresponding to the rule gets instantiated and all other corresponding variables and rules are processed.

15.1.3 PROCESSING STEPS FOR FORWARD CHAINING:

The condition variable is identified by the input from the user.
It is then placed on the conclusion variable queue for process and the variables associated with it are marked on the variable list.
The clause variable list is scanned to search for a variable name that suits exactly with the variable which is in the front of the conclusion variable queue. IF a variable is found, the rule corresponding to the variable is found using the formula,
Rule # = ((Slot #/4) + 1)*10
Next comes the instantiation part where all the variables that has to be processed get instantiated one by one and finally the THEN part gets executed.

These steps are followed repeatedly until there are no more variables in the conclusion variable queue. At this stage the final answer of forward chaining is revealed.

Forwad Chaining Variable List

1. Cancer

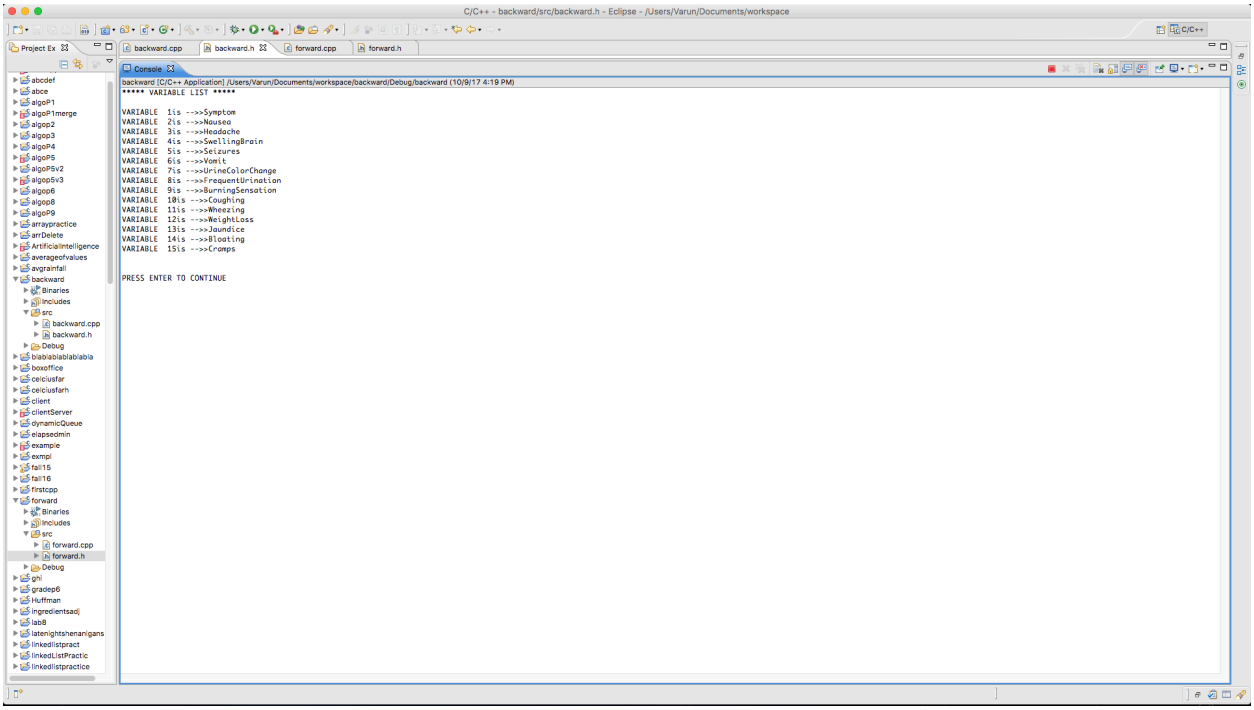
Clause Variable List :

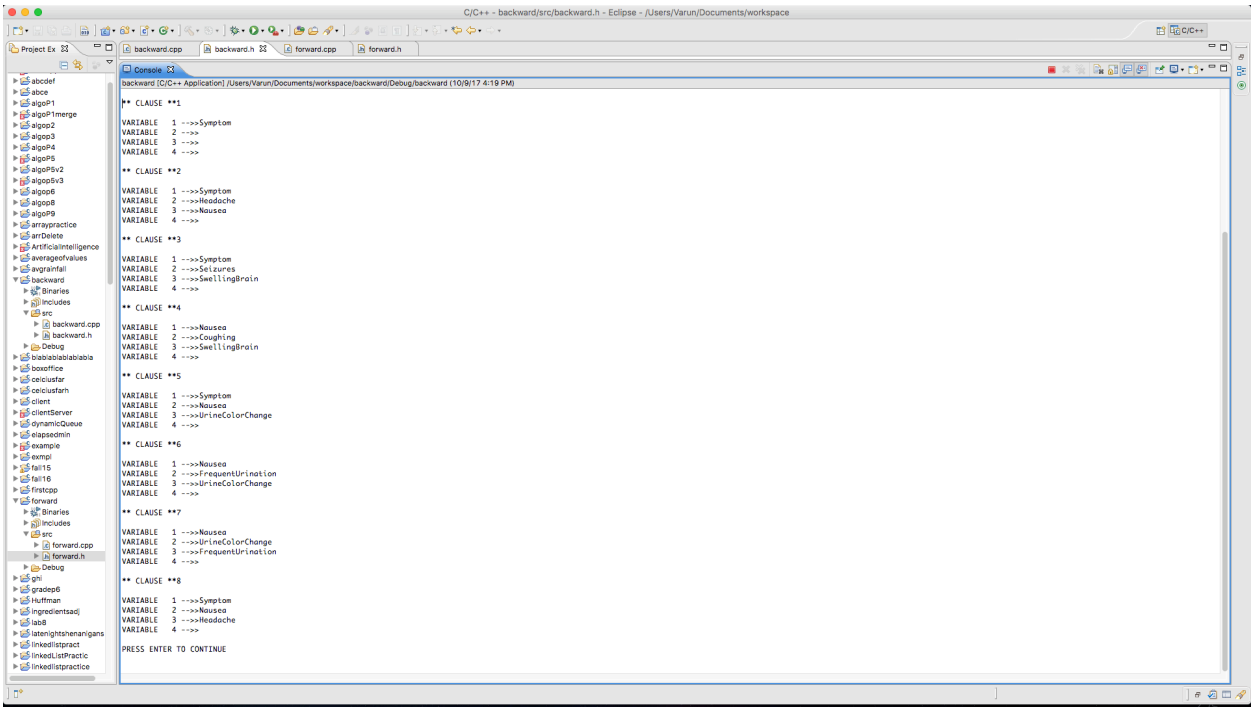
1	Cancer
2	
3	
4	
5	Cancer
6	
7	
8	
9	Cancer
10	
11	
12	
13	Cancer
14	
15	
16	
17	Cancer
18	
19	

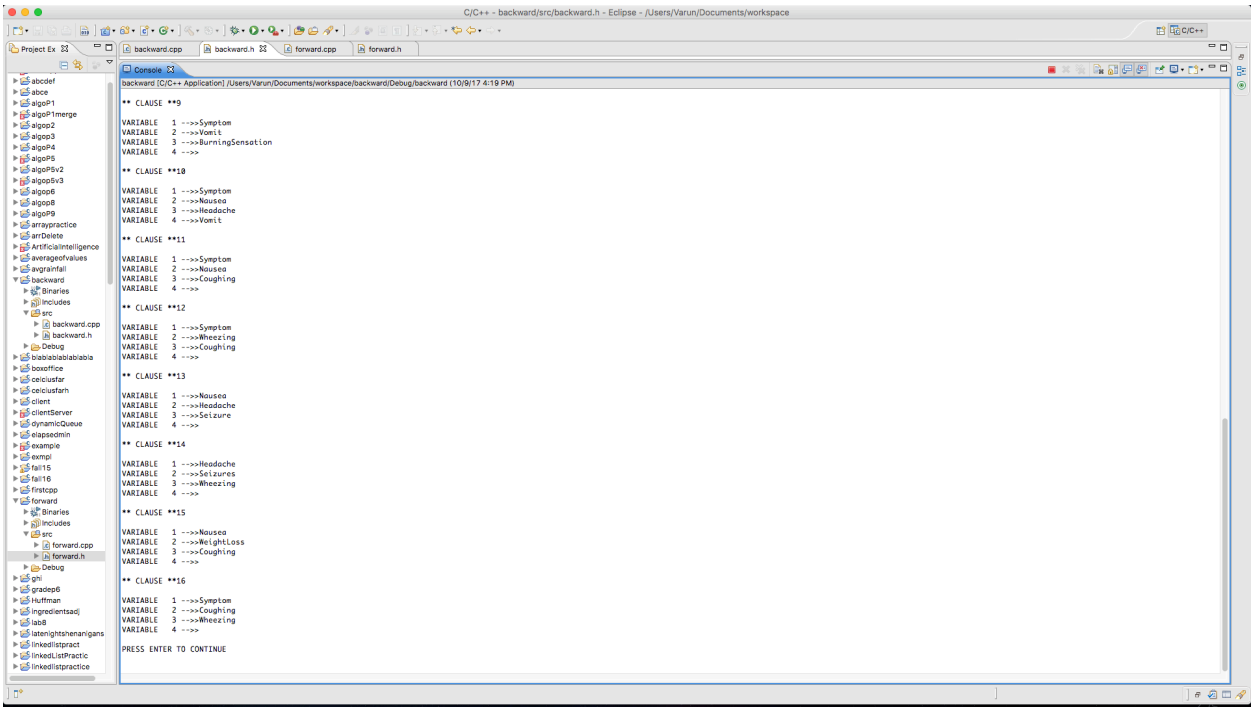
15.1.9 ScreenShots:

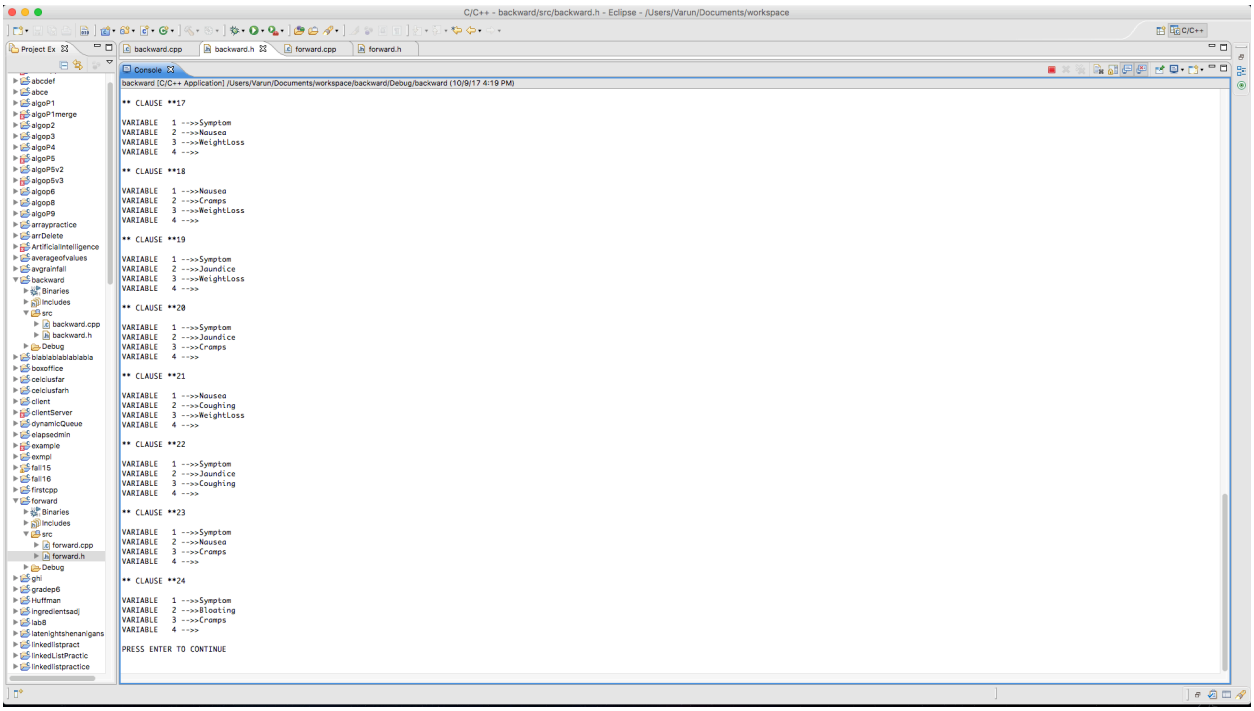
Backward Chaining

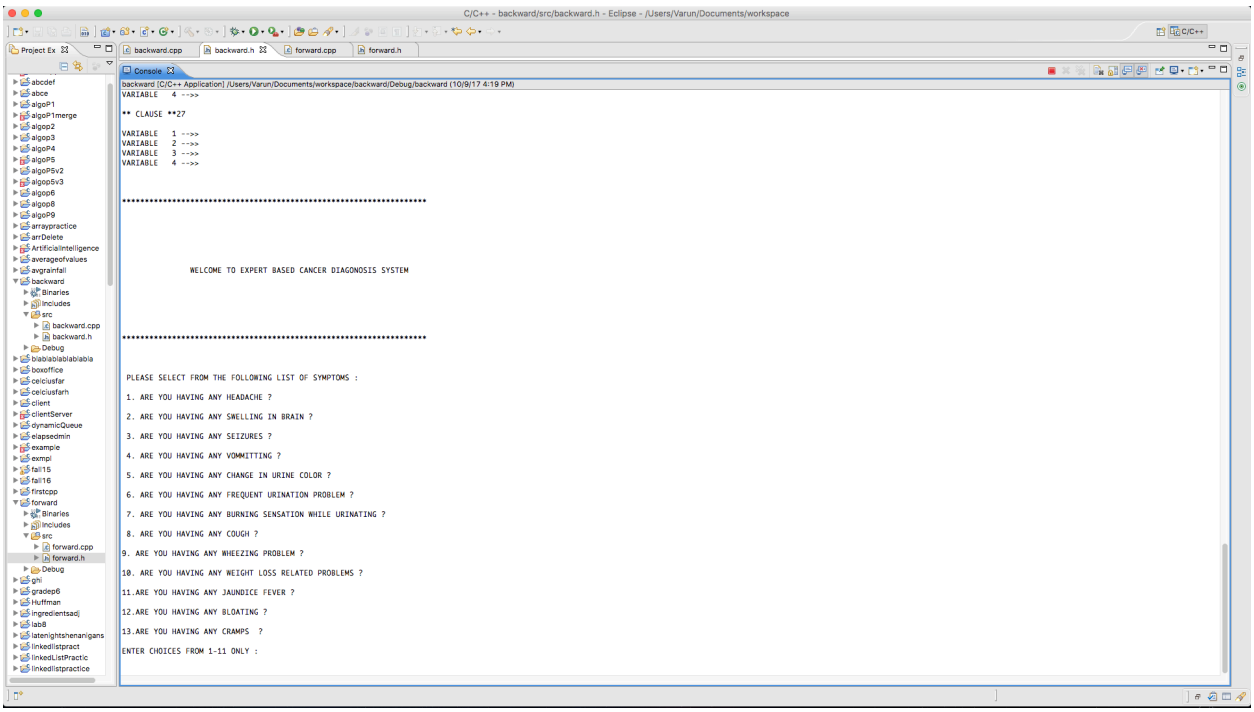
Patient 1

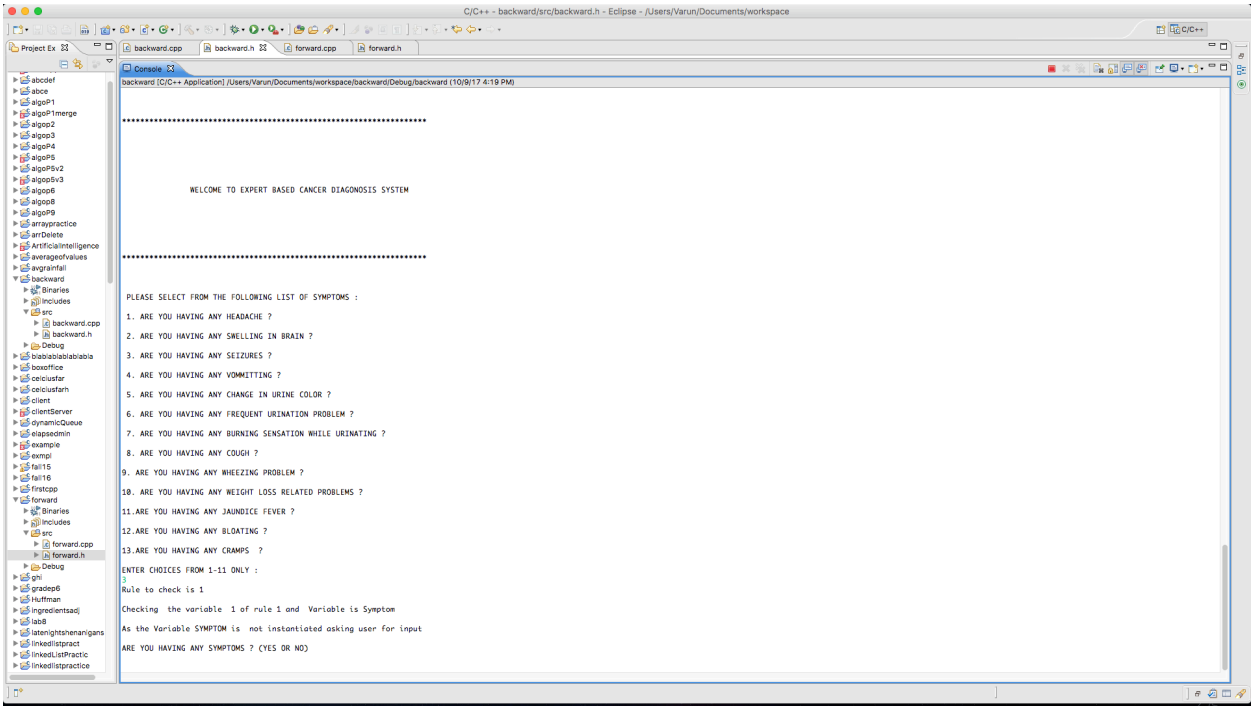


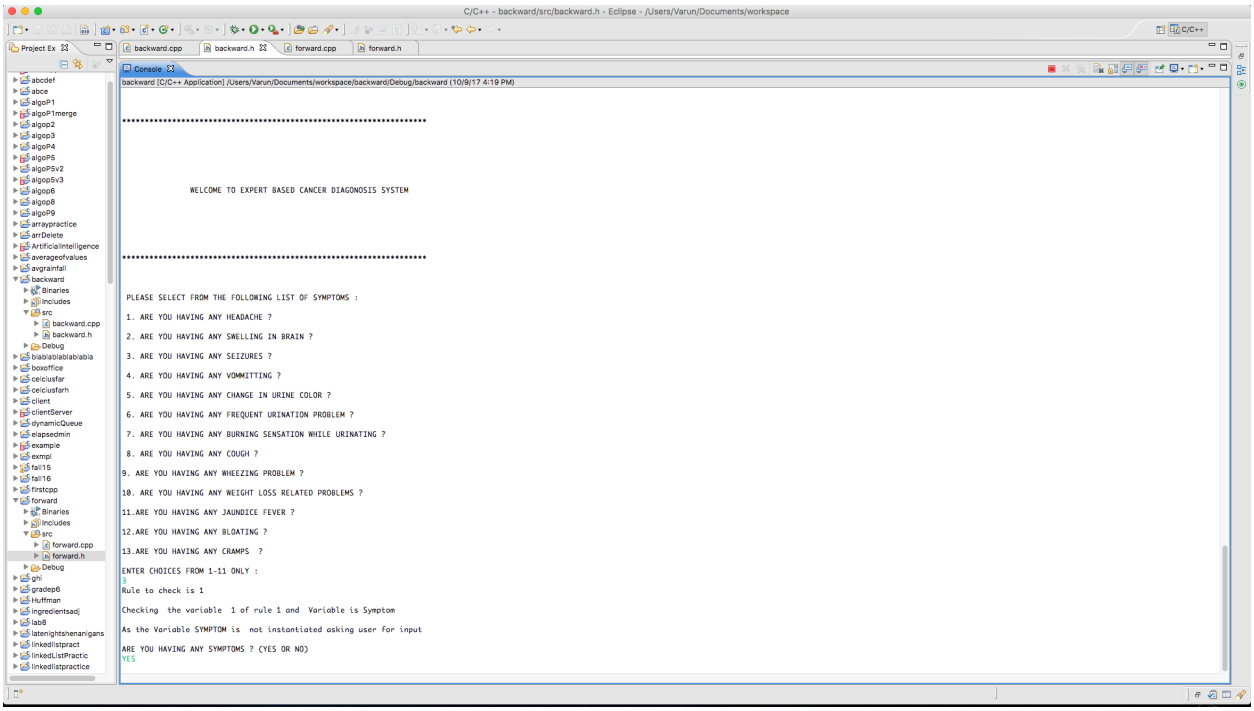


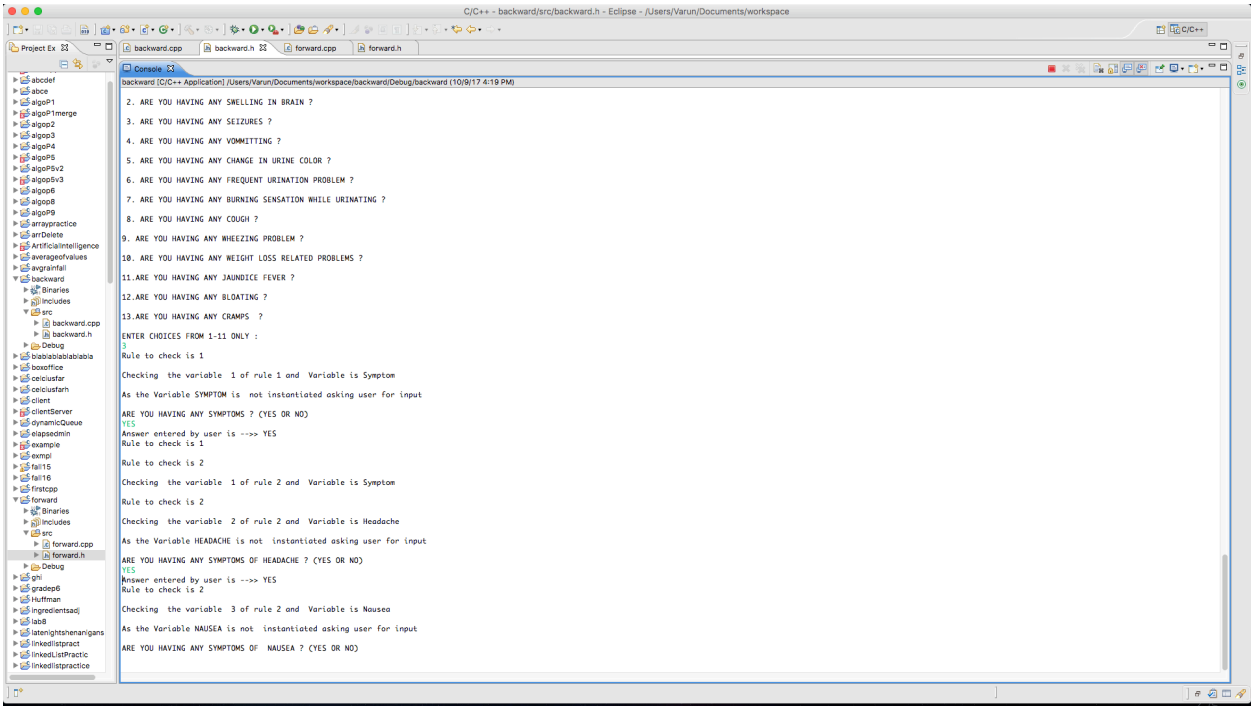

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:19 PM)
** CLAUSE **9
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Vomit
VARIABLE 3 -->>BurningSensation
VARIABLE 4 -->>
** CLAUSE **10
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Nausea
VARIABLE 3 -->>Headache
VARIABLE 4 -->>Vomit
** CLAUSE **11
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Nausea
VARIABLE 3 -->>Coughing
VARIABLE 4 -->>
** CLAUSE **12
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Wheezing
VARIABLE 3 -->>Coughing
VARIABLE 4 -->>
** CLAUSE **13
VARIABLE 1 -->>Nausea
VARIABLE 2 -->>Headache
VARIABLE 3 -->>Seizure
VARIABLE 4 -->>
** CLAUSE **14
VARIABLE 1 -->>Headache
VARIABLE 2 -->>Seizures
VARIABLE 3 -->>Wheezing
VARIABLE 4 -->>
** CLAUSE **15
VARIABLE 1 -->>Nausea
VARIABLE 2 -->>WeightLoss
VARIABLE 3 -->>Coughing
VARIABLE 4 -->>
** CLAUSE **16
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Coughing
VARIABLE 3 -->>Wheezing
VARIABLE 4 -->>
PRESS ENTER TO CONTINUE

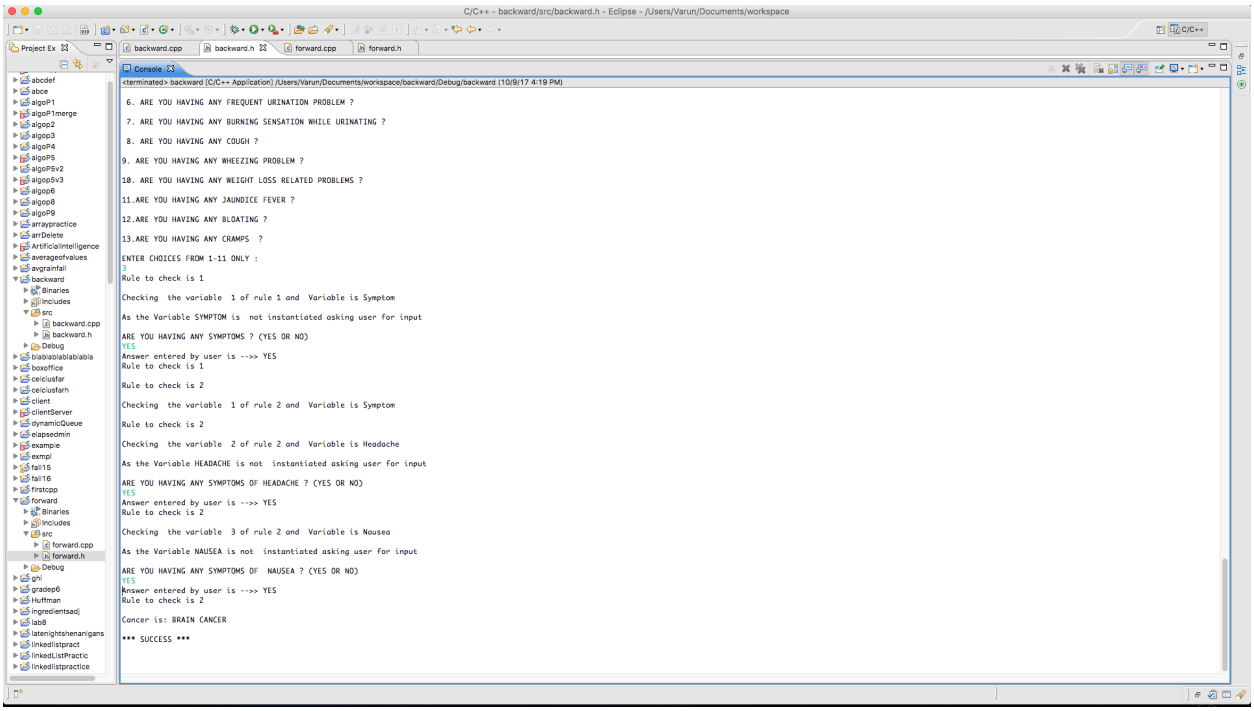

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:19 PM)
** CLAUSE **17
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Nausea
VARIABLE 3 -->>WeightLoss
VARIABLE 4 -->>
** CLAUSE **18
VARIABLE 1 -->>Nausea
VARIABLE 2 -->>Cramps
VARIABLE 3 -->>WeightLoss
VARIABLE 4 -->>
** CLAUSE **19
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Jaundice
VARIABLE 3 -->>WeightLoss
VARIABLE 4 -->>
** CLAUSE **20
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Jaundice
VARIABLE 3 -->>Cramps
VARIABLE 4 -->>
** CLAUSE **21
VARIABLE 1 -->>Nausea
VARIABLE 2 -->>Coughing
VARIABLE 3 -->>WeightLoss
VARIABLE 4 -->>
** CLAUSE **22
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Jaundice
VARIABLE 3 -->>Coughing
VARIABLE 4 -->>
** CLAUSE **23
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Nausea
VARIABLE 3 -->>Cramps
VARIABLE 4 -->>
** CLAUSE **24
VARIABLE 1 -->>Symptom
VARIABLE 2 -->>Bloating
VARIABLE 3 -->>Cramps
VARIABLE 4 -->>
PRESS ENTER TO CONTINUE


C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:19 PM)
VARIABLE 4 -->>
** CLAUSE **27
VARIABLE 1 -->>
VARIABLE 2 -->>
VARIABLE 3 -->>
VARIABLE 4 -->>
****
WELCOME TO EXPERT BASED CANCER DIAGONOSIS SYSTEM
****
PLEASE SELECT FROM THE FOLLOWING LIST OF SYMPTOMS :
1. ARE YOU HAVING ANY HEADACHE ?
2. ARE YOU HAVING ANY SWELLING IN BRAIN ?
3. ARE YOU HAVING ANY SEIZURES ?
4. ARE YOU HAVING ANY VOMMITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :

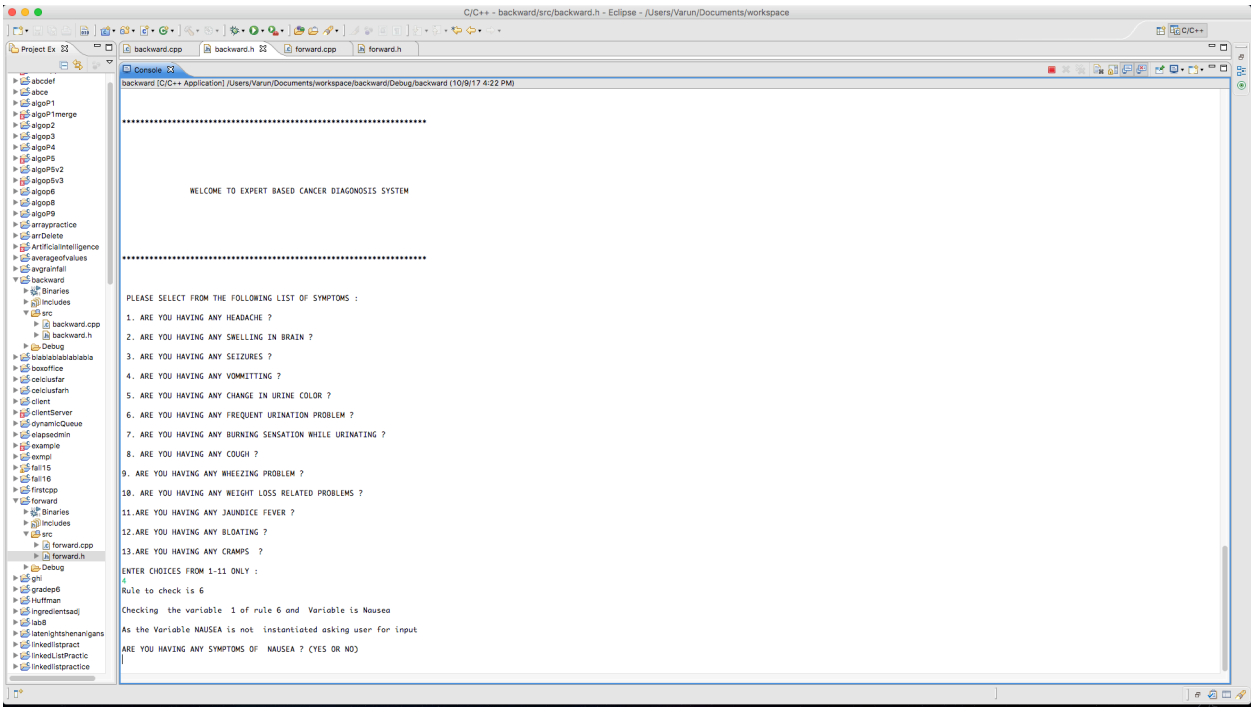

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:19 PM)
****
WELCOME TO EXPERT BASED CANCER DIAGONOSIS SYSTEM
****
PLEASE SELECT FROM THE FOLLOWING LIST OF SYMPTOMS :
1. ARE YOU HAVING ANY HEADACHE ?
2. ARE YOU HAVING ANY SWELLING IN BRAIN ?
3. ARE YOU HAVING ANY SEIZURES ?
4. ARE YOU HAVING ANY VOMMITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
3
Rule to check is 1
Checking the variable 1 of rule 1 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)

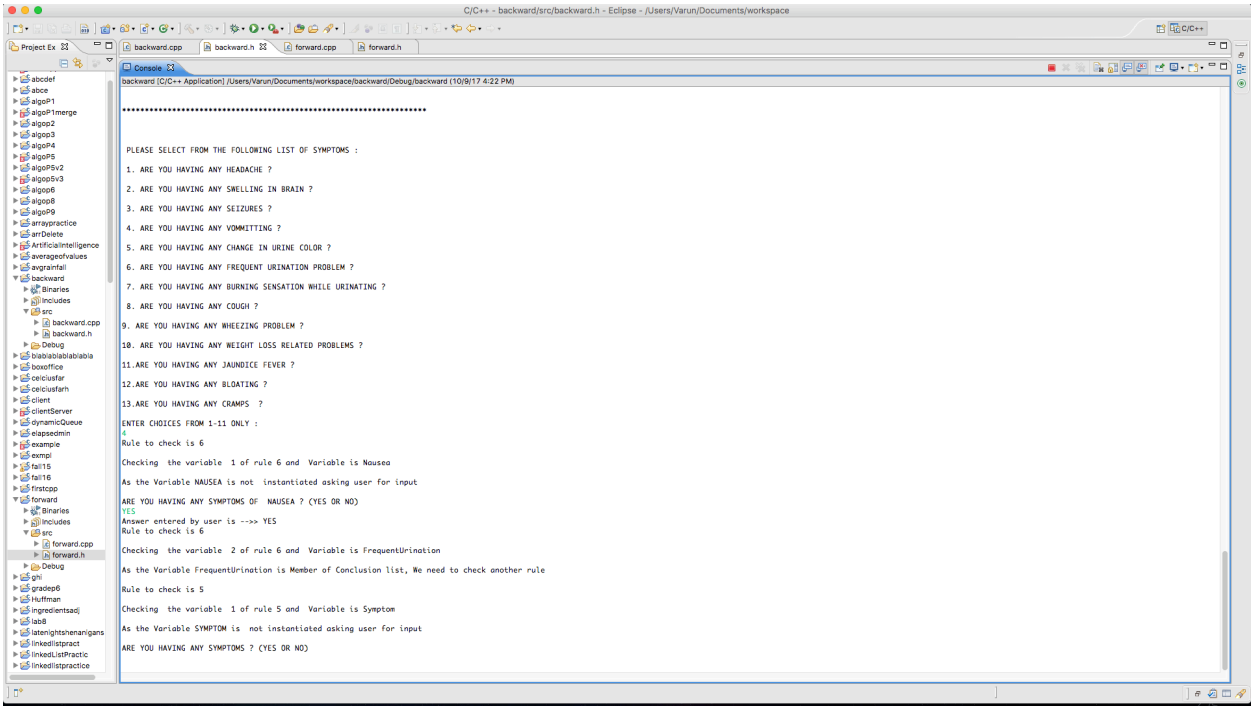

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:19 PM)
****
WELCOME TO EXPERT BASED CANCER DIAGONOSIS SYSTEM
****
PLEASE SELECT FROM THE FOLLOWING LIST OF SYMPTOMS :
1. ARE YOU HAVING ANY HEADACHE ?
2. ARE YOU HAVING ANY SWELLING IN BRAIN ?
3. ARE YOU HAVING ANY SEIZURES ?
4. ARE YOU HAVING ANY VOMMITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
3
Rule to check is 1
Checking the variable 1 of rule 1 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)
YES

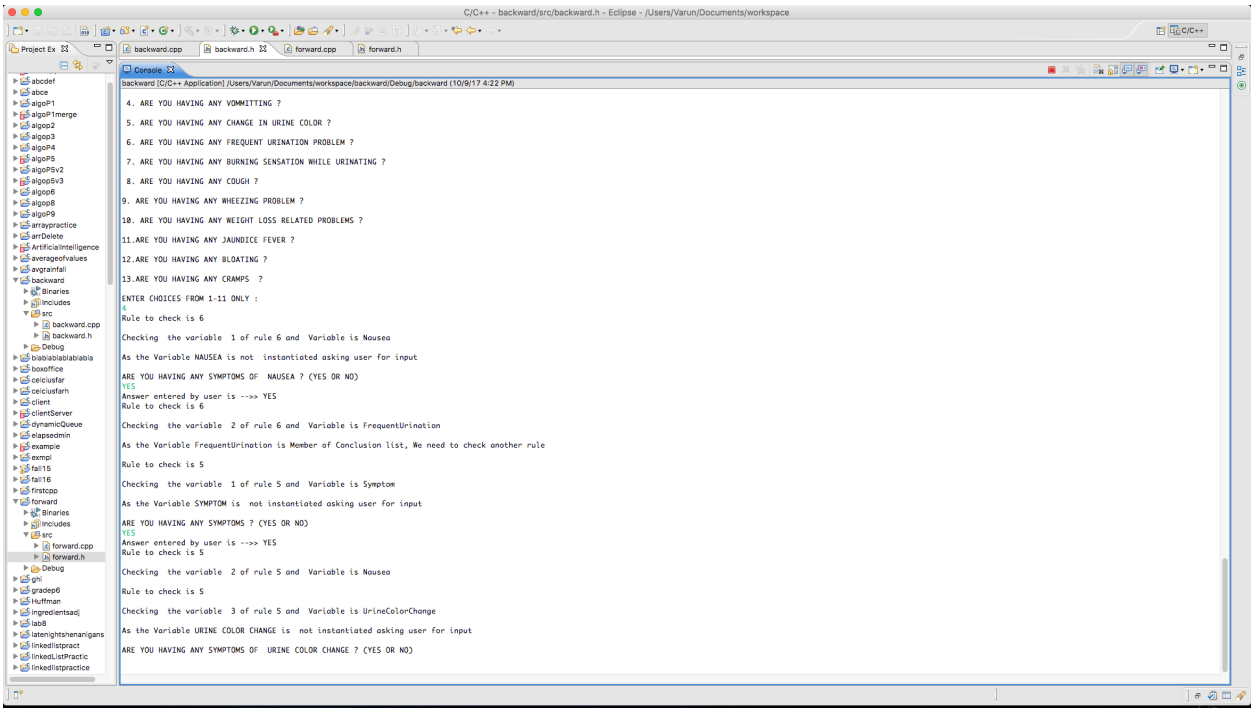

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:19 PM)
2. ARE YOU HAVING ANY SWELLING IN BRAIN ?
3. ARE YOU HAVING ANY SEIZURES ?
4. ARE YOU HAVING ANY VOMMITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
3
Rule to check is 1
Checking the variable 1 of rule 1 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 1
Rule to check is 2
Checking the variable 1 of rule 2 and Variable is Symptom
Rule to check is 2
Checking the variable 2 of rule 2 and Variable is Headache
As the Variable HEADACHE is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF HEADACHE ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 2
Checking the variable 3 of rule 2 and Variable is Nausea
As the Variable NAUSEA is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF NAUSEA ? (YES OR NO)

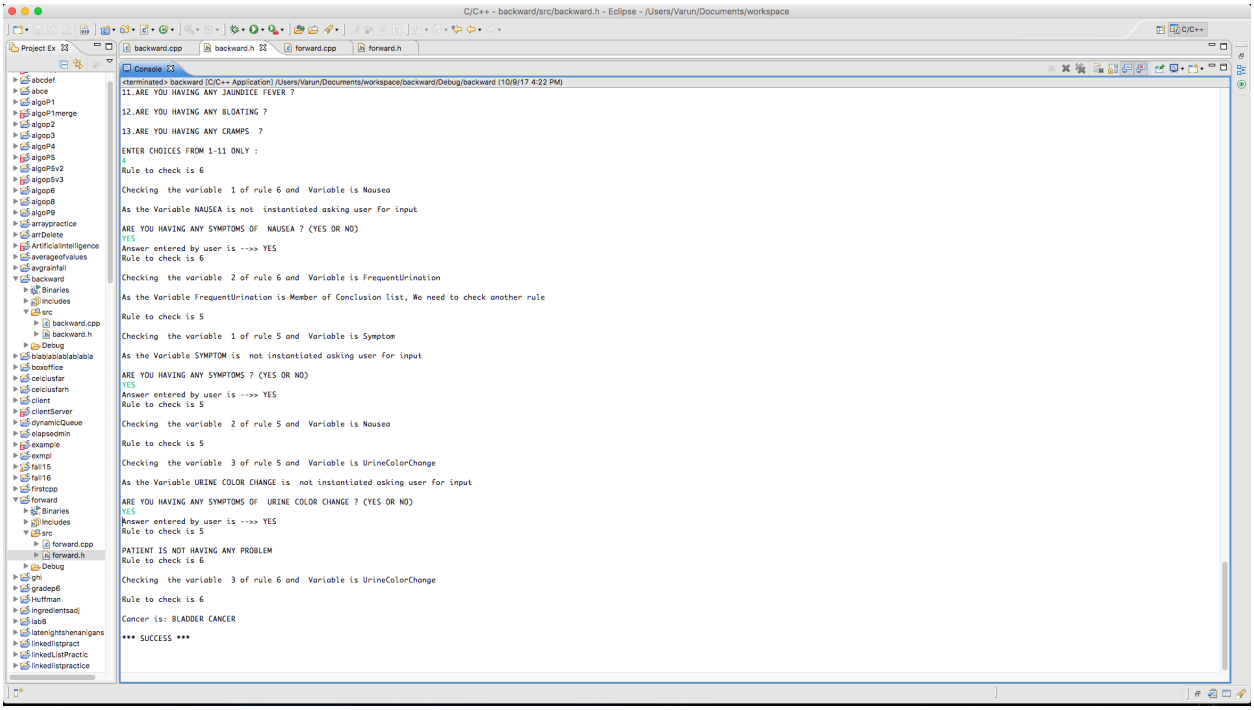

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
<terminated> backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:19 PM)
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
3
Rule to check is 1
Checking the variable 1 of rule 1 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)
YES
Answer entered by user is -->> YES
Rule to check is 1
Rule to check is 2
Checking the variable 1 of rule 2 and Variable is Symptom
Rule to check is 2
Checking the variable 2 of rule 2 and Variable is Headache
As the Variable HEADACHE is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF HEADACHE ? (YES OR NO)
YES
Answer entered by user is -->> YES
Rule to check is 2
Checking the variable 3 of rule 2 and Variable is Nausea
As the Variable NAUSEA is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF NAUSEA ? (YES OR NO)
YES
Answer entered by user is -->> YES
Rule to check is 2
Cancer is: BRAIN CANCER
*** SUCCESS ***

PATIENT-2:

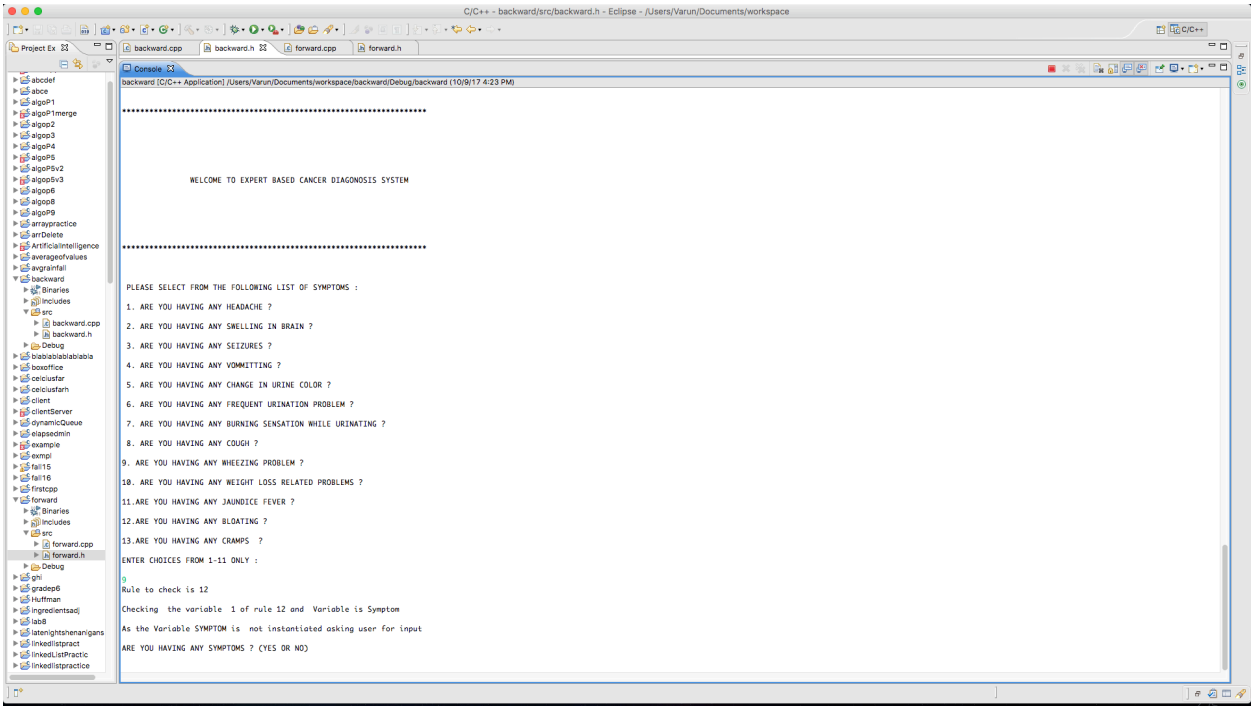

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:22 PM)
****
WELCOME TO EXPERT BASED CANCER DIAGONOSIS SYSTEM
****
PLEASE SELECT FROM THE FOLLOWING LIST OF SYMPTOMS :
1. ARE YOU HAVING ANY HEADACHE ?
2. ARE YOU HAVING ANY SWELLING IN BRAIN ?
3. ARE YOU HAVING ANY SEIZURES ?
4. ARE YOU HAVING ANY VOMMITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
4
Rule to check is 6
Checking the variable 1 of rule 6 and Variable is Nausea
As the Variable NAUSEA is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF NAUSEA ? (YES OR NO)

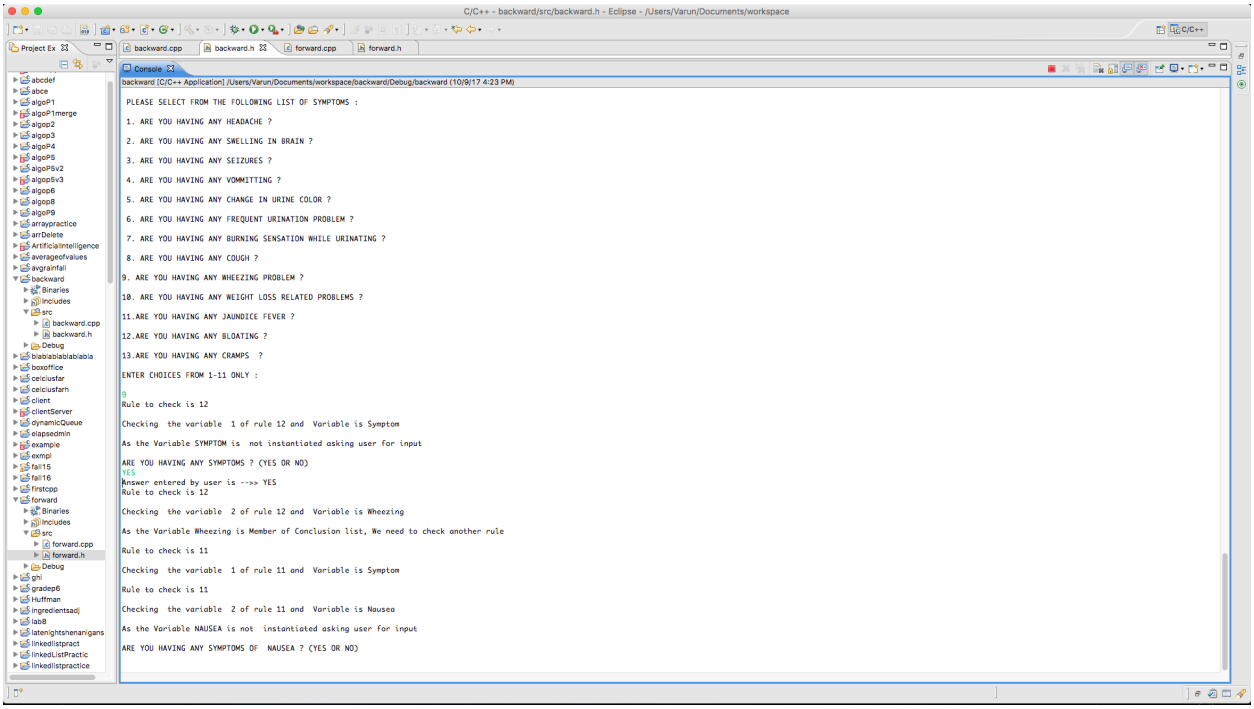

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:22 PM)
****
PLEASE SELECT FROM THE FOLLOWING LIST OF SYMPTOMS :
1. ARE YOU HAVING ANY HEADACHE ?
2. ARE YOU HAVING ANY SWELLING IN BRAIN ?
3. ARE YOU HAVING ANY SEIZURES ?
4. ARE YOU HAVING ANY VOMMITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
4
Rule to check is 6
Checking the variable 1 of rule 6 and Variable is Nausea
As the Variable NAUSEA is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF NAUSEA ? (YES OR NO)
YES
Answer entered by user is --> YES
Rule to check is 6
Checking the variable 2 of rule 6 and Variable is FrequentUrination
As the Variable FrequentUrination is Member of Conclusion list, We need to check another rule
Rule to check is 5
Checking the variable 1 of rule 5 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)

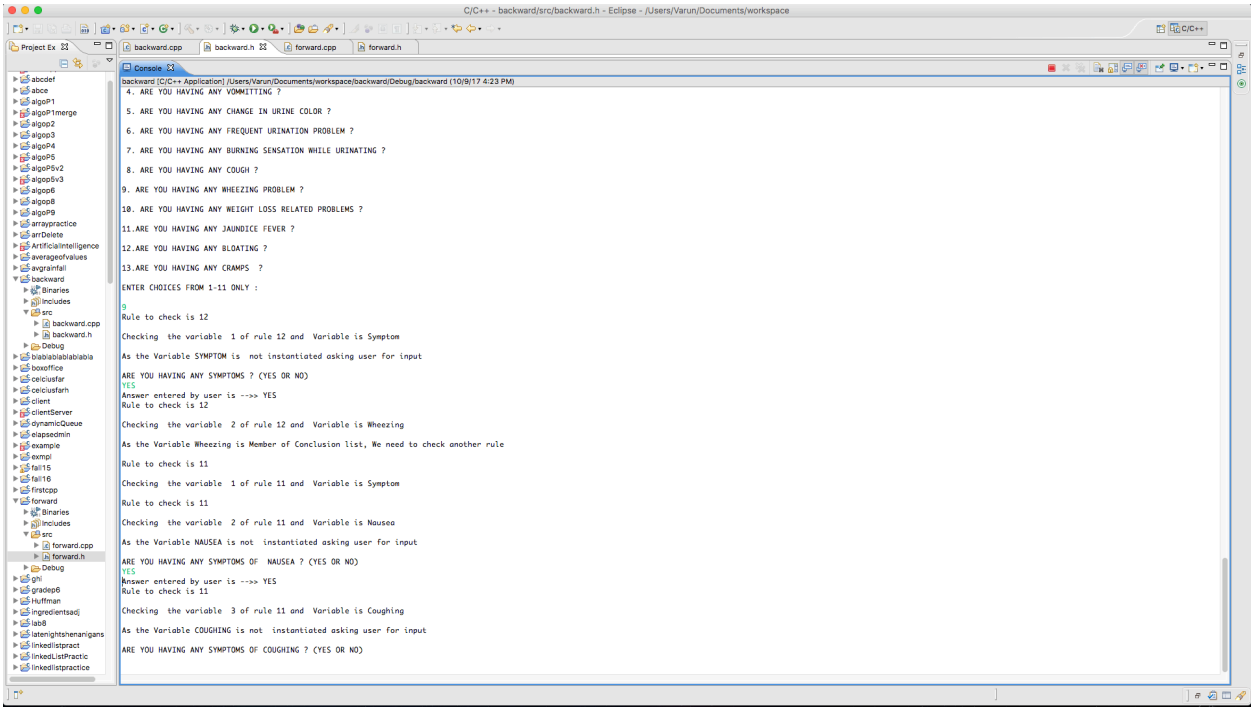

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:22 PM)
4. ARE YOU HAVING ANY VOMMITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
4
Rule to check is 6
Checking the variable 1 of rule 6 and Variable is Nausea
As the Variable NAUSEA is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF NAUSEA ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 6
Checking the variable 2 of rule 6 and Variable is FrequentUrination
As the Variable FrequentUrination is Member of Conclusion list, We need to check another rule
Rule to check is 5
Checking the variable 1 of rule 5 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 5
Checking the variable 2 of rule 5 and Variable is Nausea
Rule to check is 5
Checking the variable 3 of rule 5 and Variable is UrineColorChange
As the Variable URINE COLOR CHANGE is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF URINE COLOR CHANGE ? (YES OR NO)

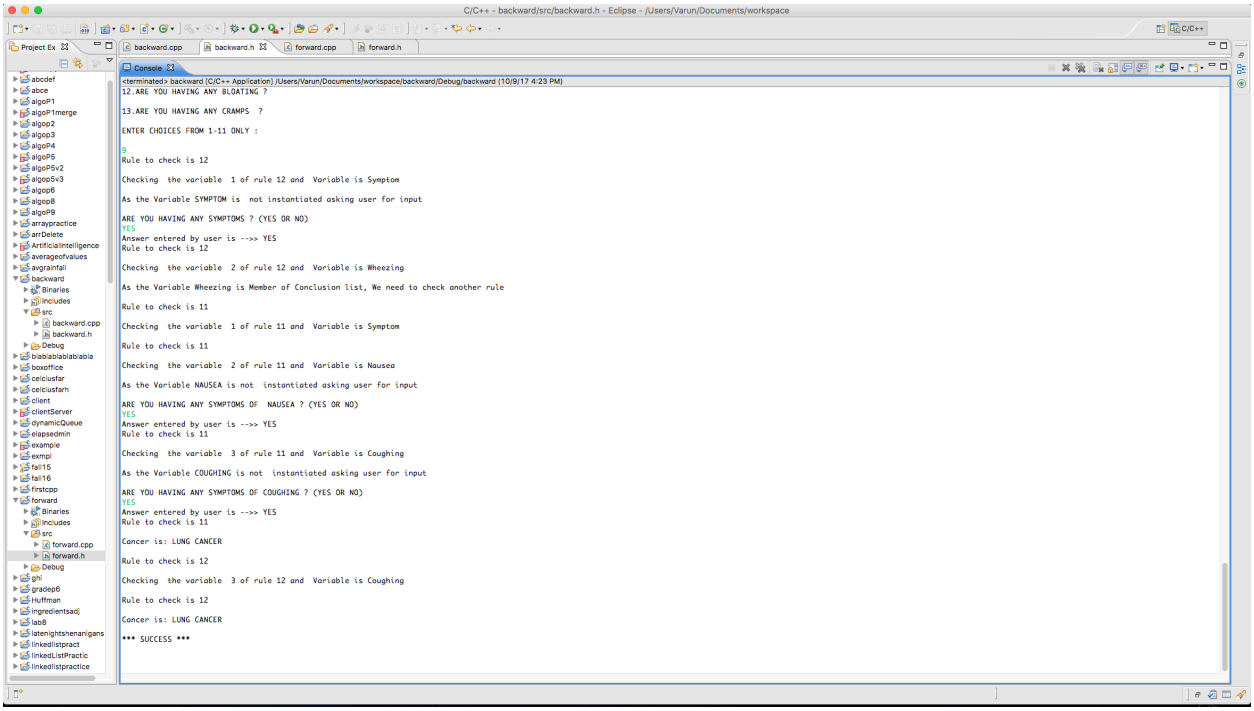

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
<terminated> backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:22 PM)
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
4
Rule to check is 6
Checking the variable 1 of rule 6 and Variable is Nausea
As the Variable NAUSEA is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF NAUSEA ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 6
Checking the variable 2 of rule 6 and Variable is FrequentUrination
As the Variable FrequentUrination is Member of Conclusion list, We need to check another rule
Rule to check is 5
Checking the variable 1 of rule 5 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 5
Checking the variable 2 of rule 5 and Variable is Nausea
Rule to check is 5
Checking the variable 3 of rule 5 and Variable is UrineColorChange
As the Variable URINE COLOR CHANGE is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF URINE COLOR CHANGE ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 5
PATIENT IS NOT HAVING ANY PROBLEM
Rule to check is 6
Checking the variable 3 of rule 6 and Variable is UrineColorChange
Rule to check is 6
Cancer is: BLADDER CANCER
*** SUCCESS ***

PATIENT-3:

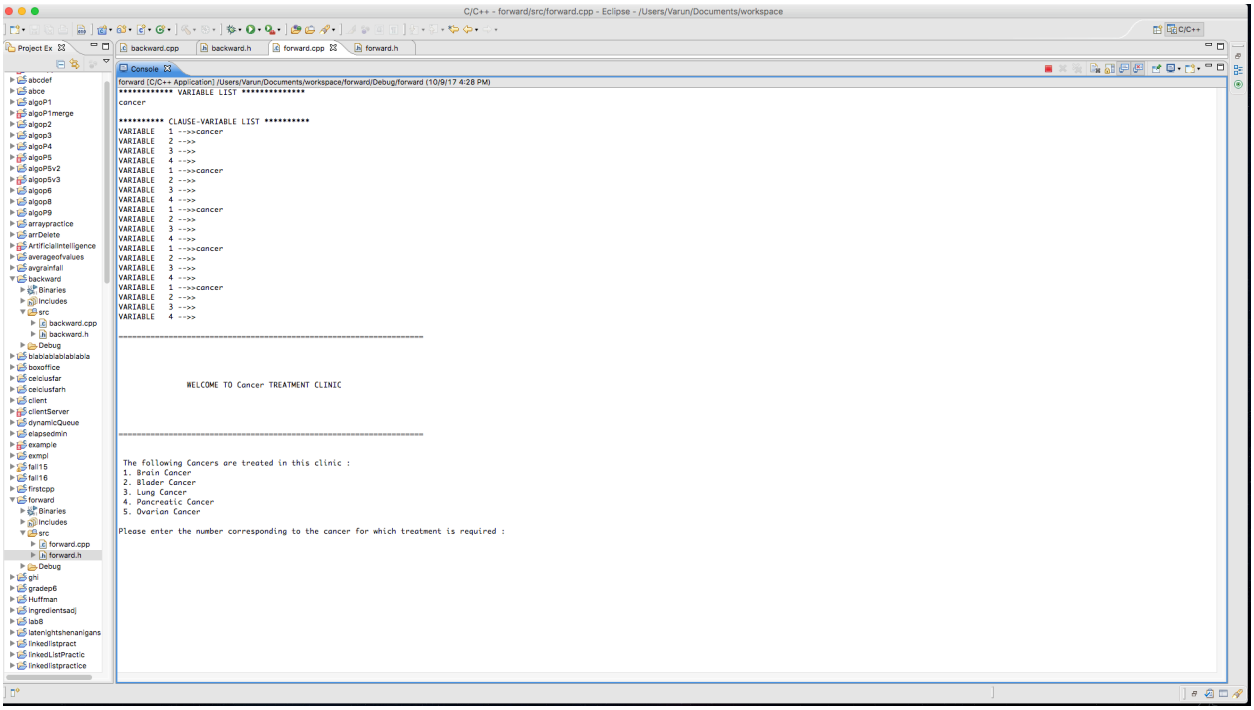

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:23 PM)
****
WELCOME TO EXPERT BASED CANCER DIAGONOSIS SYSTEM
****
PLEASE SELECT FROM THE FOLLOWING LIST OF SYMPTOMS :
1. ARE YOU HAVING ANY HEADACHE ?
2. ARE YOU HAVING ANY SWELLING IN BRAIN ?
3. ARE YOU HAVING ANY SEIZURES ?
4. ARE YOU HAVING ANY VOMMITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
9
Rule to check is 12
Checking the variable 1 of rule 12 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)

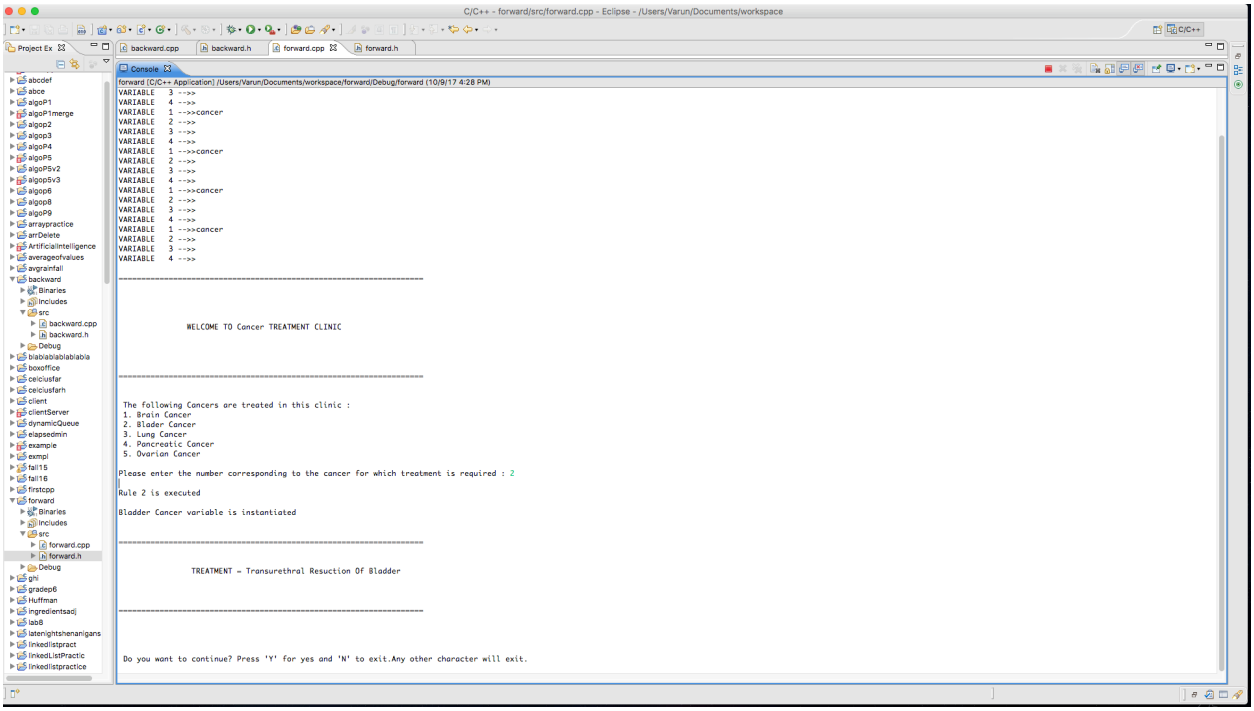

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:23 PM)
PLEASE SELECT FROM THE FOLLOWING LIST OF SYMPTOMS :
1. ARE YOU HAVING ANY HEADACHE ?
2. ARE YOU HAVING ANY SWELLING IN BRAIN ?
3. ARE YOU HAVING ANY SEIZURES ?
4. ARE YOU HAVING ANY VOMMITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
9
Rule to check is 12
Checking the variable 1 of rule 12 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 12
Checking the variable 2 of rule 12 and Variable is Wheezing
As the Variable Wheezing is Member of Conclusion list, We need to check another rule
Rule to check is 11
Checking the variable 1 of rule 11 and Variable is Symptom
Rule to check is 11
Checking the variable 2 of rule 11 and Variable is Nausea
As the Variable NAUSEA is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF NAUSEA ? (YES OR NO)

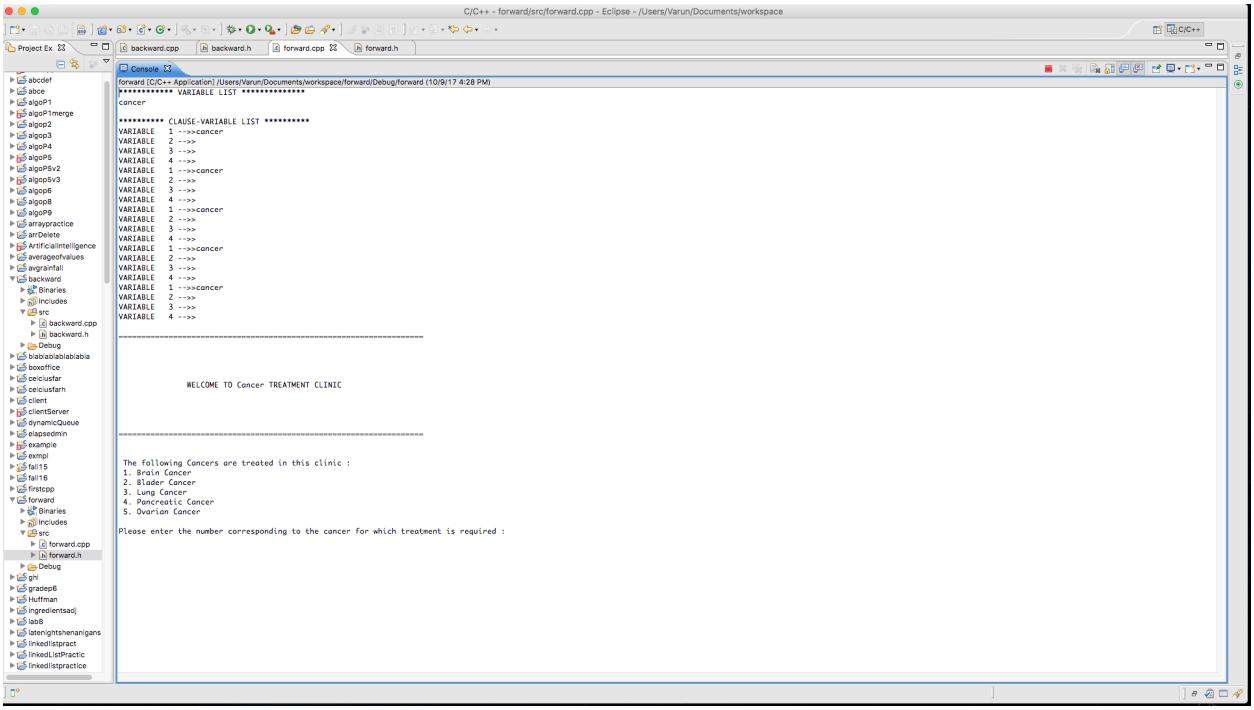

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:23 PM)
4. ARE YOU HAVING ANY VOMKITTING ?
5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
6. ARE YOU HAVING ANY FREQUENT URINATION PROBLEM ?
7. ARE YOU HAVING ANY BURNING SENSATION WHILE URINATING ?
8. ARE YOU HAVING ANY COUGH ?
9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
10. ARE YOU HAVING ANY WEIGHT LOSS RELATED PROBLEMS ?
11.ARE YOU HAVING ANY JAUNDICE FEVER ?
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
9
Rule to check is 12
Checking the variable 1 of rule 12 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 12
Checking the variable 2 of rule 12 and Variable is Wheezing
As the Variable Wheezing is Member of Conclusion list, We need to check another rule
Rule to check is 11
Checking the variable 1 of rule 11 and Variable is Symptom
Rule to check is 11
Checking the variable 2 of rule 11 and Variable is Nausea
As the Variable NAUSEA is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF NAUSEA ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 11
Checking the variable 3 of rule 11 and Variable is Coughing
As the Variable COUGHING is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF COUGHING ? (YES OR NO)

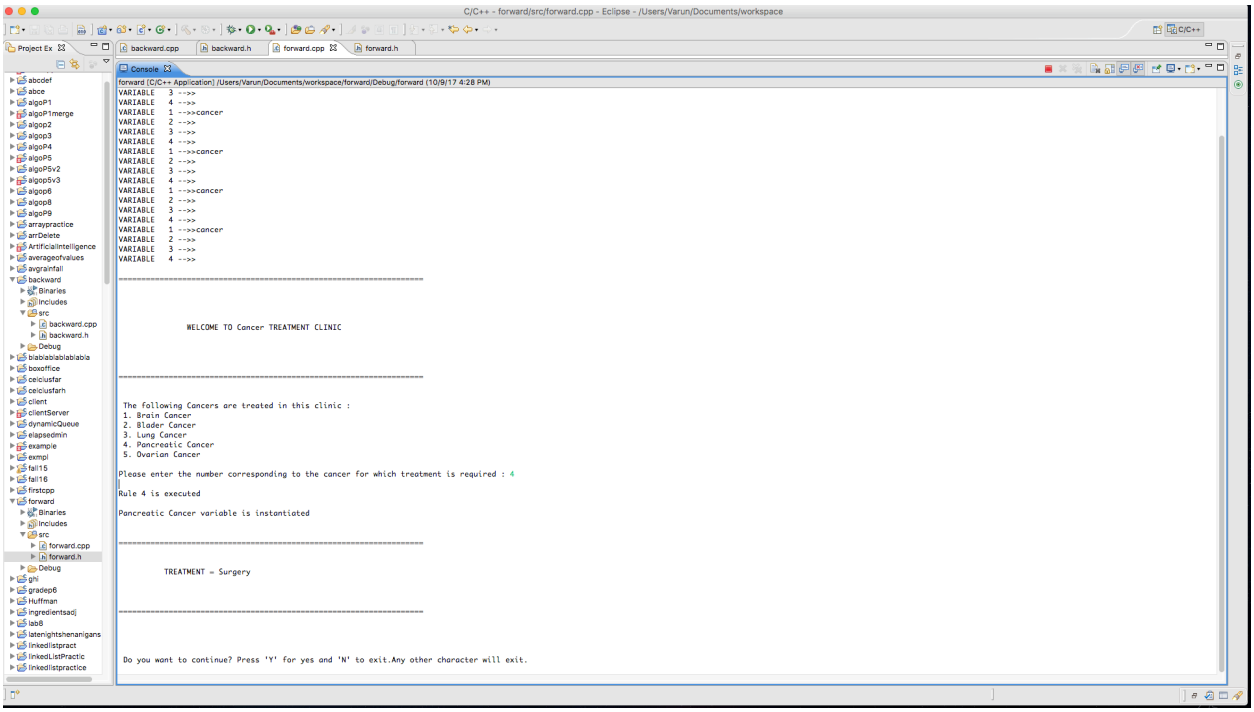

C/C++ - backward/src/backward.h - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
<terminated> backward [C/C++ Application] /Users/Varun/Documents/workspace/backward/Debug/backward (10/9/17 4:23 PM)
12.ARE YOU HAVING ANY BLOATING ?
13.ARE YOU HAVING ANY CRAMPS ?
ENTER CHOICES FROM 1-11 ONLY :
9
Rule to check is 12
Checking the variable 1 of rule 12 and Variable is Symptom
As the Variable SYMPTOM is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 12
Checking the variable 2 of rule 12 and Variable is Wheezing
As the Variable Wheezing is Member of Conclusion list, We need to check another rule
Rule to check is 11
Checking the variable 1 of rule 11 and Variable is Symptom
Rule to check is 11
Checking the variable 2 of rule 11 and Variable is Nausea
As the Variable NAUSEA is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF NAUSEA ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 11
Checking the variable 3 of rule 11 and Variable is Coughing
As the Variable COUGHING is not instantiated asking user for input
ARE YOU HAVING ANY SYMPTOMS OF COUGHING ? (YES OR NO)
YES
Answer entered by user is --->> YES
Rule to check is 11
Cancer is: LUNG CANCER
Rule to check is 12
Checking the variable 3 of rule 12 and Variable is Coughing
Rule to check is 12
Cancer is: LUNG CANCER
*** SUCCESS ***

Forward Chaining






C/C++ - forward/src/forward.cpp - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
forward [C/C++ Application] /Users/Varun/Documents/workspace/forward/Debug/forward (10/9/17 4:28 PM)
********** VARIABLE LIST **********
cancer
********** CLAUSE-VARIABLE LIST **********
VARIABLE 1 -->>cancer
VARIABLE 2 -->>
VARIABLE 3 -->>
VARIABLE 4 -->>
VARIABLE 1 -->>cancer
VARIABLE 2 -->>
VARIABLE 3 -->>
VARIABLE 4 -->>
VARIABLE 1 -->>cancer
VARIABLE 2 -->>
VARIABLE 3 -->>
VARIABLE 4 -->>
VARIABLE 1 -->>cancer
VARIABLE 2 -->>
VARIABLE 3 -->>
VARIABLE 4 -->>
VARIABLE 1 -->>cancer
VARIABLE 2 -->>
VARIABLE 3 -->>
VARIABLE 4 -->>
WELCOME TO Cancer TREATMENT CLINIC
The following Cancers are treated in this clinic :
1. Brain Cancer
2. Bladder Cancer
3. Lung Cancer
4. Pancreatic Cancer
5. Ovarian Cancer
Please enter the number corresponding to the cancer for which treatment is required :


C/C++ - forward/src/forward.cpp - Eclipse - /Users/Varun/Documents/workspace
backward.cpp
backward.h
forward.cpp
forward.h
Console
forward [C/C++ Application] /Users/Varun/Documents/workspace/forward/Debug/forward (10/9/17 4:28 PM)
VARIABLE 3 -->>
VARIABLE 4 -->>
VARIABLE 1 -->>cancer
VARIABLE 2 -->>
VARIABLE 3 -->>
VARIABLE 4 -->>
VARIABLE 1 -->>cancer
VARIABLE 2 -->>
VARIABLE 3 -->>
VARIABLE 4 -->>
VARIABLE 1 -->>cancer
VARIABLE 2 -->>
VARIABLE 3 -->>
VARIABLE 4 -->>
WELCOME TO Cancer TREATMENT CLINIC
The following Cancers are treated in this clinic :
1. Brain Cancer
2. Bladder Cancer
3. Lung Cancer
4. Pancreatic Cancer
5. Ovarian Cancer
Please enter the number corresponding to the cancer for which treatment is required : 4
Rule 4 is executed
Pancreatic Cancer variable is instantiated
TREATMENT = Surgery
Do you want to continue? Press 'Y' for yes and 'N' to exit.Any other character will exit.

15.2 BACKWARD CHAINING (GOAL DRIVEN) ALGORITHM AND IMPLEMENTATION:

Backward chaining starts with a list of goals and works backwards from the consequent to the antecedent to check whether the data is available that supports these consequents. Inference engine using this backward chaining would search the inference rules until it finds one which has a consequent that matches a desired goal. If the antecedent (IF clause) of that rule is not known to be true, then it is added to the list of goals.

15.2.1 PROCESSING THE KNOWLEDGE BASE:

The knowledge base of backward chaining begins with creating variable name table and framing of rules based on those variable names.

15.2.2 DATA STRUCTURES USED IN BACKWARD CHAINING:

Clause Variable List:

Clause variable list stores all the variables used in IF part of the rules. These variables are stored in array with four array slots allocated for each slot. If only one or two array slots are filled, the remaining slots are left blank. If all the clauses in the IF part of a rule are connected by the logical operator AND, all the variables in these clauses must be instantiated before the THEN part can be executed.

Conclusion Stack:

Conclusion stack is the most important data structure in implementing backward chaining. It tells which rule contains the conclusion that we are trying to reach and which clause number in the IF portion is currently examined for instantiated.

Conclusion List:

A clause variable pointer keeps track of the current rule and currently executed clause in that rule. It consists of rule number, conclusion associated with that rule number and set of conditions which yields the conclusion. Conclusion list is complete when the THEN portion of each rule is placed in the same row as the rule number. If the IF part of a rule is true, we invoke the THEN part and instantiate the conclusion.

Variable List:

This structure contains two items: one is a variable name for each variable contained in the IF part of the knowledge base rules and the other item tells us whether or not the variable is instantiated. A variable only appears once in the list no matter how many condition clauses it appears. The instantiated column is always initially set to not instantiated (NI). It will be changed to instantiated (I) as each variable is set to a value.

15.2.3 PROCESSING STEPS FOR BACKWARD CHAINING:

The conclusion for the problem is identified.

Search 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.

Formula to find slot #:

Slot # = ((Rule #/10) - 1) *4) + 1

Instantiate the IF clause of the statement.

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.

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.

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.

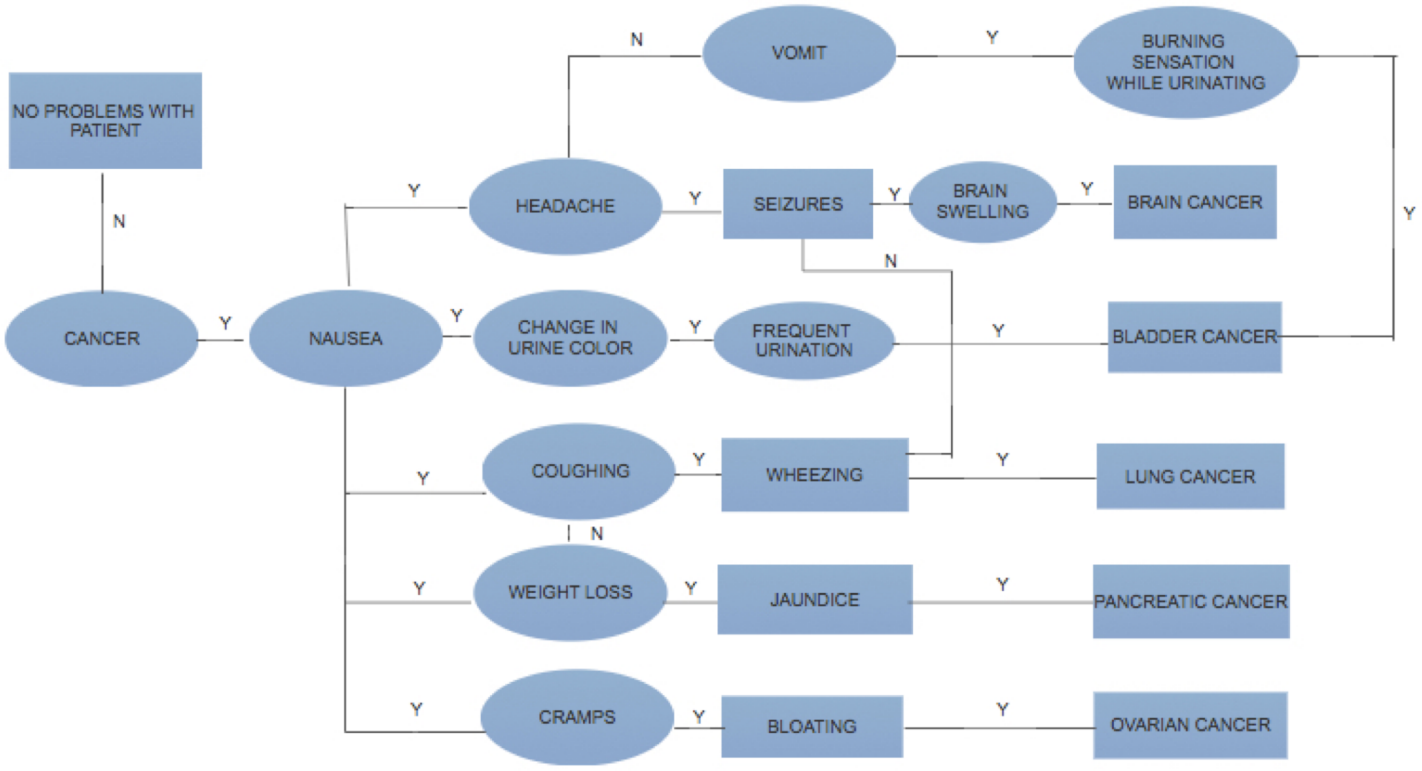
If such a statement is found, go back to step 3.

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, then go back to step 6.

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.

15.2.3 Decision Tree FOR BACKWARD CHAINING:



BACKWARD CHAINING- DECISION TREE

15.2.4 RULES FOR BACKWARD AND FORWARDS CHAINING:

BACKWARD CHAINING :

10. if(Cancer==yes && Nausea==yes && headache==yes)
    then seizures=yes

20. if(seizures==yes && brainswelling==yes)
    then BrainCancer=yes

30. if(Cancer==yes && Nausea==yes && urinecolorchange==yes)
    then frequenturination=yes

40. if(Nausea==yes && urinecolorchange==yes && frequenturination==yes)
    then BladderCancer=yes

50. if(frequenturination==yes && urinecolorchange==)
    then BladderCancer=yes

60. if(Nausea==yes && Headache==no s)
    then Vomit=yes

70. if(Vomit==yes && burningsensation== s)
    then BladderCancer=yes

80. if(Nausea==yes && urinecolorchange==yes && frequenturination==no)
    then BladderCancer=no

90. if(Nausea==yes && headache==yes && Vomit==no)
    then BladderCancer=no

100. if(Cancer==yes && Nausea==yes && Coughing==yes)
    then wheezing=yes

110. if(wheezing==yes && Coughing==yes)
    then LungCancer=yes

120. if(Nausea==yes && headache==yes && seizure==no)
then wheezing=yes

130. if(headache==yes && seizure==no && wheezing==yes)
then LungCancer=yes

140. if(Nausea==yes && weightloss==no && coughing==yes)
then wheezing=yes

150. if(coughing==yes && wheezing==yes)
then LungCancer=yes

160. if(Cancer==yes && nausea==yes && weightloss==yes)
then jaundice=yes

170. if(jaundice==yes && weightloss==yes)
then PancreaticCancer=yes

180. if(Nausea==yes && coughing==yes && weightloss==yes)
then jaundice=yes

190. if(jaundice==yes && coughing==yes)
then PancreaticCancer=yes

200. if(Nausea==yes && cramps==yes && weightloss==yes)
then jaundice=yes

210. if(jaundice==yes && cramps==yes)
then PancreaticCancer=yes

220. if(Cancer==yes && nausea==yes && cramps==yes)
then bloating=yes

230. if(bloating==yes && cramps==)
then OvarianCancer=yes

240. if(Cancer==yes && nausea==yes && bloating==no)
then Cancer=no

250. if(Cancer==no)
then Cancer=no

Forward chaining rules

10.

If (cancer == BrainCancer)

Then Treatment = Chemo Therapy

20.

If (cancer == BladderCancer)

Then Treatment = Transurethral Resuction Of Bladder

30.

If (cancer == LungCancer)

Then Treatment = Radiation

40.

If (cancer == PancreaticCancer)

Then Treatment = Surgery

50.

If (cancer == OvarianCancer)

Then Treatment = Surgery and Chemo Therapy

Clause variable list

In our program we allocate room for four variables each rule

Rule #	Clause Variable Name
1	cancer
2	
3	
4	
5	cancer
6	nausea
7	headache
8	
9	cancer
10	seizures
11	swellingbrain
12	
13	cancer
14	nausea
15	coughing
16	swellingbrain
17	cancer
18	nausea
19	urinecolorchange
20	
21	cancer
22	frequenturination
23	urinecolorchange
24	
25	cancer
26	nausea
27	headache
28	
29	cancer
30	vomit
31	Burningsensation

32	
33	cancer
34	nausea
35	headache
36	
37	cancer
38	nausea
39	coughing
40	
41	cancer
42	wheezing
43	coughing
44	
45	cancer
46	nausea
47	headache
48	seizures
49	cancer
50	headache
51	seizures
52	wheezing
53	cancer
54	nausea
55	weightloss
56	coughing
57	cancer

58	coughing
59	wheezing
60	
61	cancer
62	nausea
63	weightloss
64	
65	cancer
66	jaundice
67	weightloss
68	
69	cancer
70	nausea
71	cramps
72	weightloss
73	cancer
74	jaundice
75	cramps
76	
77	cancer
78	jaundice
79	coughing
80	
81	symptom
82	nausea
83	cramps

84	bloating
85	cancer
86	bloating
87	cramps

Conclusion List

10	Cancer
20	Seizures
30	Jaundice
40	Wheezing
50	Bloating
60	BrainCancer
70	BladderCancer
80	LungCancer
90	PancreaticCancer

Variable List

1	Cancer
2	Nausea
3	Headache
4	SwellingInBrain
5	Seizures
6	Vomit
7	UrineColorChange
8	FrequentUrination
9	BurningSensation
10	Coughing
11	Wheezing

12	WeightLoss
13	Jaundice
14	Bloating
15	Cramps

16.Source Code:

```
//Sidhant Chadha and Varun Dalal
#include "backward.h"
#include <iostream>
#include <stdio.h>
#include<stdlib.h>
#include <string.h>
#include<cstdlib>
#include<cstring>

using namespace std;


int main()

{
intializing();
Display();


cout<<"\n"<<endl;
cout<<"**************************************************
**********"<<endl;
cout<<"\n\n\n\n\n"<<endl;
cout<<"               WELCOME TO EXPERT BASED CANCER
DIAGONOSIS SYSTEM      "<<endl;
```

```
cout<<"\n\n\n\n\n\n"<<endl;

cout<<"**************************************************
**********\n\n"<<endl;

cout<<"\n PLEASE SELECT FROM THE FOLLOWING LIST OF
SYMPTOMS : \n"<<endl;

cout<<" 1. ARE YOU HAVING ANY HEADACHE ? \n"<<endl;
cout<<" 2. ARE YOU HAVING ANY SWELLING IN BRAIN ?
\n"<<endl;
cout<<" 3. ARE YOU HAVING ANY SEIZURES ?  \n"<<endl;
cout<<" 4. ARE YOU HAVING ANY VOMMITTING ?  \n"<<endl;
cout<<" 5. ARE YOU HAVING ANY CHANGE IN URINE COLOR ?
\n"<<endl;
cout<<" 6. ARE YOU HAVING ANY FREQUENT URINATION
PROBLEM ? \n"<<endl;
cout<<" 7. ARE YOU HAVING ANY BURNING SENSATION WHILE
URINATING ?  \n"<<endl;
cout<<" 8. ARE YOU HAVING ANY COUGH ? \n"<<endl;
cout<<"9. ARE YOU HAVING ANY WHEEZING PROBLEM ?
\n"<<endl;
cout<<"10. ARE YOU HAVING ANY WEIGHT LOSS RELATED
PROBLEMS ? \n"<<endl;
cout<<"11.ARE YOU HAVING ANY JAUNDICE FEVER ?
\n"<<endl;
cout<<"12.ARE YOU HAVING ANY BLOATING ? \n"<<endl;
cout<<"13.ARE YOU HAVING ANY CRAMPS  ? \n"<<endl;


/****** inference section *****/
char choice[1];
renter:
cout<<"ENTER CHOICES FROM 1-11 ONLY : "<<endl;
```

```
cin>>choice;
if(strcmp(choice,"1")== 0)
strcpy(Variable,"BrainCancer");

else if(strcmp(choice,"2")== 0)
strcpy(Variable,"BrainCancer");

else if(strcmp(choice,"3")==0)
strcpy(Variable,"BrainCancer");

else if(strcmp(choice,"4")==0)
strcpy(Variable,"BladderCancer");

else if(strcmp(choice,"5")==0)
strcpy(Variable,"BladderCancer");

else if(strcmp(choice,"6")==0)
strcpy(Variable,"BladderCancer");

else if(strcmp(choice,"7")==0)
strcpy(Variable,"BladderCancer");

else if(strcmp(choice,"8")==0)
strcpy(Variable,"LungCancer");

else if(strcmp(choice,"9")==0)
strcpy(Variable,"LungCancer");

else if(strcmp(choice,"10")==0)
strcpy(Variable,"PancreaticCancer");

else if(strcmp(choice,"11")==0)
strcpy(Variable,"PancreaticCancer");
```

```
else if(strcmp(choice,"12")==0)
strcpy(Variable,"OvarianCancer");

else if(strcmp(choice,"13")==0)
strcpy(Variable,"OvarianCancer");



else
goto renter;

/* Getting Conclusion from user and finding out the
conclusion number from the conclusion list */
b520: f=1;
determine_member_concl_list();
int rule;
if (sn != 0)
{
/***pushing the statement no and clause number  on
stack and incrementing the stack**/
do
{

push_on_stack();
do
{
/* calculating  clause location in clause-variable list
*/

b545: i= (statsk[sp] -1) *4
+ clausk[sp];

rule=statsk[sp];
cout<<"Rule to check is "<<statsk[sp]<<"\n"<<endl;
```

```
strcpy(Variable, ClauseVariable_List[i]);

if(strcmp(Variable,"") != 0)
{

/*determining whether it is this clause variable a
conclusion or not ? */
f = 1;
cout<<"Checking  the variable  "<<clausk[sp]<<" of rule
"<<rule<<" and  Variable is "<<Variable<<"\n"<<endl;
determine_member_concl_list();
if(sn != 0)
{

/* Means conclusion again pushing statement no  and
conclusion no on  stack */
cout<<"As the Variable " <<Variable<<" is Member of
Conclusion list, We need to check another rule
\n"<<endl;
goto b520;


}

instantiate();

clausk[sp] = clausk[sp] + 1;

}

}while(strcmp(Variable, "") != 0);
```

```
/*no more clauses check if part of statement */


sn = statsk[sp];


s = 0;
//cout<<"Sn : "<<sn;

switch (sn) {


case 1: if ((strcmp(Symptom, "NO") == 0))


s = 1;

break;


case 2:// cout<<"S = "<<Symptom<<" H = "<<Headache<<" N
= "<<Nausea;
if((strcmp(Symptom, "YES") == 0) &&
strcmp(Headache, "YES") == 0 && strcmp(Nausea,
"YES")==0)
{
  strcpy(Seizures,"YES");
  s = 1;
 // cout<<"In LOOP";
}

break;
```

```
case 3: if((strcmp(Symptom, "YES") == 0) &&
strcmp(Seizures, "YES") == 0 && strcmp(SwellingBrain,
"YES")==0 )
{
  strcpy(BrainCancer,"YES");
  s = 1;
}
break;

case 4 : if((strcmp(Symptom, "YES") == 0) &&
/* strcmp(Nausea, "YES") == 0 && */strcmp(Coughing,
"YES")==0 && strcmp(SwellingBrain, "YES")==0 )
{
  strcpy(BrainCancer,"YES");
  s = 1;
}



break;
case 5: if((strcmp(Symptom, "YES") == 0) &&
strcmp(Nausea, "YES") == 0 && strcmp(UrineColorChange,
"YES")==0 )
{
//  cout<<"Symptom"<<Symptom<<endl>;
//  cout<<""
  strcpy(FrequentUrination,"YES");
  s = 1;
}
break;

break;

case 6 : if((strcmp(Symptom, "YES") == 0) &&
```

```
strcmp(FrequentUrination, "YES") == 0 &&
strcmp(UrineColorChange, "YES")==0 )

{
  strcpy(BladderCancer,"YES");
  s = 1;
}
break;


case 7 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Nausea, "YES") == 0 && strcmp(UrineColorChange,
"YES")==0) /*&& strcmp(FrequentUrination, "NO")==0 )*/

{
  strcpy(BladderCancer,"YES");
  s = 1;
}
break;
case 8 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Nausea, "YES") == 0 && strcmp(Headache,
"NO")==0)

{
  strcpy(Vomit,"YES");
  s = 1;
}
break;
case 9 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Vomit, "YES") == 0 && strcmp(BurningSensation,
"YES")==0)

{
  strcpy(BladderCancer,"YES");
```

```
  s = 1;
}
break;
case 10 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Nausea, "YES") == 0 && strcmp(Headache,
"YES")==0 && strcmp(Vomit, "NO")==0 )

{
  strcpy(BladderCancer,"NO");
  s = 1;
}
break;
case 11 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Nausea, "YES") == 0 && strcmp(Coughing,
"YES")==0)

{
  strcpy(Wheezing,"YES");
  s = 1;
}
break;
case 12 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Wheezing, "YES") == 0 && strcmp(Coughing,
"YES")==0 )

{
  strcpy(LungCancer,"YES");
  s = 1;
}
break;
case 13 : if(
strcmp(Nausea, "YES") == 0 && strcmp(Headache,
"YES")==0 && strcmp(Seizures, "NO")==0 )
```

```
{
  strcpy(Wheezing,"YES");
  s = 1;
}
break;
case 14 : if(
strcmp(Headache, "YES") == 0 && strcmp(Seizures,
"NO")==0 && strcmp(Wheezing, "YES")==0 )

{
  strcpy(LungCancer,"YES");
  s = 1;
}

break;

case 15 : if(
strcmp(Nausea, "YES") == 0 && strcmp(WeightLoss,
"NO")==0 && strcmp(Coughing, "YES")==0 )

{
  strcpy(Wheezing,"YES");
  s = 1;
}
break;

case 16 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Coughing, "YES") == 0 && strcmp(Wheezing,
"YES")==0)

{
  strcpy(LungCancer,"YES");
  s = 1;
}
```

```
break;

case 17 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Nausea, "YES") == 0 && strcmp(WeightLoss,
"YES")==0)

{
  strcpy(Jaundice,"YES");
  s = 1;
}
break;
case 18 : if(
strcmp(Nausea, "YES") == 0 && strcmp(Cramps, "YES")==0
&& strcmp(WeightLoss, "YES")==0 )

{
  strcpy(Jaundice,"YES");
  s = 1;
}
break;
case 19 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Jaundice, "YES") == 0 && strcmp(WeightLoss,
"YES")==0 )

{
  strcpy(PancreaticCancer,"YES");
  s = 1;
}
break;
case 20 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Jaundice, "YES") == 0 && strcmp(Cramps,
"YES")==0 )

{
```

```
  strcpy(PancreaticCancer,"YES");
  s = 1;
}
break;
case 21 : if(
strcmp(Nausea, "YES") == 0 && strcmp(Coughing,
"YES")==0 && strcmp(WeightLoss, "YES")==0 )

{
  strcpy(Jaundice,"YES");
  s = 1;
}
break;
case 22 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Jaundice, "YES") == 0 && strcmp(Coughing,
"YES")==0)

{
  strcpy(PancreaticCancer,"YES");
  s = 1;
}
break;

case 23 : if((strcmp(Symptom, "YES") == 0) &&
strcmp(Nausea, "YES") == 0 && strcmp(Bloating,
"YES")==0)

{
  strcpy(OvarianCancer,"YES");
  s = 1;
}
break;
case 24 : if((strcmp(Symptom, "YES") == 0) &&
```

```
strcmp(Bloating, "YES") == 0 && strcmp(Cramps,
"YES")==0 )

{
  strcpy(OvarianCancer,"YES");
  s = 1;
}
break;
case 25 : if(
strcmp(Nausea, "YES") == 0 && strcmp(Cramps, "YES")==0
&& strcmp(Bloating, "NO")==0 )

{
  strcpy(OvarianCancer,"NO");
  s = 1;
}
break;









}
```

```
if( s != 1)

{

/* failed..search rest of statements for same
conclusion */

/* get conclusion */
i = statsk[sp];

strcpy(Variable, Conclusion_List[i]);

/* search for conclusion starting at the next statement
number */

f = statsk[sp] + 1;

determine_member_concl_list();

sp = sp+1;



}
/* pop old conclusion and put on new one */
}while((s != 1) && (sn !=0));


if(s==0 && sn ==0 && sp !=42)
```

```
cout<<"\n\n ******************** PROBLEM NOT FOUND IN
DATABASE   ********************\n\n "<<s<<endl;
if(s != 0)

{
switch (sn) {

case 1: strcpy(Symptom, "NO");

cout<<"Patient is not having any kind of Cancer.
\n"<<endl;

break;

case 2: strcpy(Seizures, "YES");

cout<<"Cancer is: BRAIN CANCER \n"<<endl;

break;


case 3: strcpy(BrainCancer, "YES");

cout<<"Cancer is: BRAIN CANCER \n"<<endl;
break;


case 4: strcpy(BrainCancer, "YES");

cout<<"Cancer is: BRAIN CANCER \n"<<endl;
break;


case 5: strcpy(FrequentUrination, "YES");
```

```
cout<<"PATIENT IS NOT HAVING ANY PROBLEM "<<endl;

break;



case 6: strcpy(BladderCancer, "YES");

cout<<"Cancer is: BLADDER CANCER \n"<<endl;

break;



case 7: strcpy(BladderCancer, "YES");

cout<<"Cancer is BLADDER CANCER.\n"<<endl;

break;
case 8: strcpy(Vomit, "YES");
//  cout<<"Mental Disorder for patient is
MentalDisorderMania.\n"<<endl;
break;
case 9: strcpy(BladderCancer, "YES");
cout<<"Cancer is: BLADDER CANCER \n"<<endl;
break;
case 10: strcpy(BladderCancer, "NO");
cout<<"Patient is not having any kind of Cancer .
\n"<<endl;
break;
case 11: strcpy(Wheezing, "YES");
cout<<"Cancer is: LUNG CANCER \n"<<endl;
break;
```

```
case 12: strcpy(LungCancer, "YES");
cout<<"Cancer is: LUNG CANCER \n"<<endl;
break;
case 13: strcpy(Wheezing, "YES");
        cout<<"Patient is not having any kind of Cancer
\n"<<endl;
break;
case 14: strcpy(LungCancer, "YES");
cout<<"Cancer is: LUNG CANCER.\n"<<endl;
break;
case 15: strcpy(Wheezing, "YES");
cout<<"Cancer is: LUNG CANCER \n"<<endl;


break;
case 16: strcpy(LungCancer, "YES");
cout<<"Cancer is: LUNG CANCER  \n"<<endl;
break;
case 17: strcpy(Jaundice, "YES");
break;
case 18: strcpy(Jaundice, "YES");
break;
case 19: strcpy(PancreaticCancer, "YES");
cout<<"Cancer is: PANCREATIC CANCER  \n"<<endl;
break;
case 20: strcpy(PancreaticCancer, "YES");
cout<<"Cancer is: PANCREATIC CANCER \n"<<endl;
break;
case 21: strcpy(Jaundice, "YES");
break;
case 22: strcpy(PancreaticCancer, "YES");
cout<<"Cancer is: PANCREATIC CANCER  \n"<<endl;
break;
case 23: strcpy(Bloating, "YES");
```

```
cout<<"Cancer is: OVARIAN CANCER"<<endl;
break;
case 24: strcpy(OvarianCancer, "YES");
cout<<"Cancer is: OVARIAN CANCER \n"<<endl;
break;
case 25: strcpy(OvarianCancer, "NO");
cout<<"PATIENT IS NOT HAVING ANY PROBLEM .\n"<<endl;
break;



}

/* pop the stack */

sp=sp+1;

if(sp >= 40)

/* finished */

cout<<"*** SUCCESS *** \n"<<endl;

else {

/* stack is not empty */

/* get next clause then continue */

// cout<<"inside else of not success"<<endl;

clausk[sp] = clausk[sp]+1;

goto b545;
```

```
}

}

}

return 0;

}
```

```
//Sidhant Chadha and Varun Dalal
#define BACKWARD_H_
#include <iostream>
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
using namespace std;
char Conclusion_List[30][30];

char Variable_List[30][30];

char ClauseVariable_List[125][30];

char Variable[50];

char w;
```

```
char Cancer[4];
char BrainCancer[4];
char BladderCancer[4];
char LungCancer[4];
char PancreaticCancer[4];
char OvarianCancer[4];

char Symptom[4];
char Nausea[4];
char Headache[4];
char SwellingBrain[4];
char Seizures[4];
char Vomit[4];
char UrineColorChange[4];
char FrequentUrination[4];
char BurningSensation[4];
char Coughing[4];
char Wheezing[4];
char WeightLoss[4];
char Jaundice[4];
char Bloating[4];
char Cramps[4];



char buff[128];
int instlt[40];

int statsk[41];

int clausk[31];
int sn, f, i, j, s, k;

int sp;
```

```
void intializing()
{
sp=40;
for (i=1; i<=10; i++)
{
strcpy(Conclusion_List[i],"");

strcpy(Variable_List[i],"");

instlt[i]=0;

statsk[i]=0;

clausk[i]=0;

}

/*****Intializing Conclusion list    */

for (i=1; i<30; i++)
strcpy(ClauseVariable_List[i],"");

strcpy(Conclusion_List[1], "BrainCancer");

strcpy(Conclusion_List[2], "BrainCancer");



strcpy(Conclusion_List[3], "BrainCancer");
strcpy(Conclusion_List[4], "BrainCancer");
strcpy(Conclusion_List[5], "FrequentUrination");
```

```
strcpy(Conclusion_List[6], "BladderCancer");

strcpy(Conclusion_List[7], "BladderCancer");

strcpy(Conclusion_List[8], "Vomit");

strcpy(Conclusion_List[9], "BladderCancer");
strcpy(Conclusion_List[10], "BladderCancer");
strcpy(Conclusion_List[11], "Wheezing");
strcpy(Conclusion_List[12], "LungCancer");
strcpy(Conclusion_List[13], "Wheezing");
strcpy(Conclusion_List[14], "LungCancer");
strcpy(Conclusion_List[15], "Wheezing");
strcpy(Conclusion_List[16], "LungCancer");
strcpy(Conclusion_List[17], "Jaundice");
strcpy(Conclusion_List[18], "Jaundice");
strcpy(Conclusion_List[19], "PancreaticCancer");
strcpy(Conclusion_List[20], "PancreaticCancer");
strcpy(Conclusion_List[21], "Jaundice");
strcpy(Conclusion_List[22], "PancreaticCancer");
strcpy(Conclusion_List[23], "Bloating");
strcpy(Conclusion_List[24], "OvarianCancer");
strcpy(Conclusion_List[25], "OvarianCancer");



/****** Intializing variable list ****/



strcpy(Variable_List[1], "Symptom");
```

```
strcpy(Variable_List[2], "Nausea");

strcpy(Variable_List[3], "Headache");

strcpy(Variable_List[4], "SwellingBrain");

strcpy(Variable_List[5], "Seizures");

strcpy(Variable_List[6], "Vomit");


strcpy(Variable_List[7], "UrineColorChange");

strcpy(Variable_List[8], "FrequentUrination");

strcpy(Variable_List[9], "BurningSensation");

strcpy(Variable_List[10], "Coughing");

strcpy(Variable_List[11], "Wheezing");

strcpy(Variable_List[12], "WeightLoss");

strcpy(Variable_List[13], "Jaundice");

strcpy(Variable_List[14], "Bloating");

strcpy(Variable_List[15], "Cramps");


/*****  Clause variable List *****/
```

```
strcpy(ClauseVariable_List[1], "Symptom");

strcpy(ClauseVariable_List[5], "Symptom");
strcpy(ClauseVariable_List[6], "Headache");
strcpy(ClauseVariable_List[7], "Nausea");

strcpy(ClauseVariable_List[9], "Symptom");
strcpy(ClauseVariable_List[10], "Seizures");
strcpy(ClauseVariable_List[11], "SwellingBrain");

strcpy(ClauseVariable_List[13], "Nausea");
strcpy(ClauseVariable_List[14], "Coughing");
strcpy(ClauseVariable_List[15], "SwellingBrain");


strcpy(ClauseVariable_List[17], "Symptom");
strcpy(ClauseVariable_List[18], "Nausea");
strcpy(ClauseVariable_List[19], "UrineColorChange");

strcpy(ClauseVariable_List[21], "Nausea");
strcpy(ClauseVariable_List[22], "FrequentUrination");
strcpy(ClauseVariable_List[23], "UrineColorChange");

strcpy(ClauseVariable_List[25], "Nausea");
strcpy(ClauseVariable_List[26], "UrineColorChange");
strcpy(ClauseVariable_List[27], "FrequentUrination");



strcpy(ClauseVariable_List[29], "Symptom");
strcpy(ClauseVariable_List[30], "Nausea");
strcpy(ClauseVariable_List[31], "Headache");

strcpy(ClauseVariable_List[33], "Symptom");
```

```
strcpy(ClauseVariable_List[34], "Vomit");
strcpy(ClauseVariable_List[35], "BurningSensation");

strcpy(ClauseVariable_List[37], "Symptom");
strcpy(ClauseVariable_List[38], "Nausea");
strcpy(ClauseVariable_List[39], "Headache");
strcpy(ClauseVariable_List[40], "Vomit");



strcpy(ClauseVariable_List[41], "Symptom");
strcpy(ClauseVariable_List[42], "Nausea");
strcpy(ClauseVariable_List[43], "Coughing");

strcpy(ClauseVariable_List[45], "Symptom");
strcpy(ClauseVariable_List[46], "Wheezing");
strcpy(ClauseVariable_List[47], "Coughing");
//           strcpy(ClauseVariable_List[48],
"Seizures");

strcpy(ClauseVariable_List[49], "Nausea");
strcpy(ClauseVariable_List[50], "Headache");
strcpy(ClauseVariable_List[51], "Seizure");
// strcpy(ClauseVariable_List[52], "Seizures");

strcpy(ClauseVariable_List[53], "Headache");
strcpy(ClauseVariable_List[54], "Seizures");
strcpy(ClauseVariable_List[55], "Wheezing");
//   strcpy(ClauseVariable_List[56], "Wheezing");

strcpy(ClauseVariable_List[57], "Nausea");
strcpy(ClauseVariable_List[58], "WeightLoss");
strcpy(ClauseVariable_List[59], "Coughing");
//     strcpy(ClauseVariable_List[60], "Coughing");
```

```
strcpy(ClauseVariable_List[61], "Symptom");
strcpy(ClauseVariable_List[62], "Coughing");
strcpy(ClauseVariable_List[63], "Wheezing");

strcpy(ClauseVariable_List[65], "Symptom");
strcpy(ClauseVariable_List[66], "Nausea");
strcpy(ClauseVariable_List[67], "WeightLoss");

strcpy(ClauseVariable_List[69], "Nausea");
strcpy(ClauseVariable_List[70], "Cramps");
strcpy(ClauseVariable_List[71], "WeightLoss");
//   strcpy(ClauseVariable_List[72], "WeightLoss");

strcpy(ClauseVariable_List[73], "Symptom");
strcpy(ClauseVariable_List[74], "Jaundice");
strcpy(ClauseVariable_List[75], "WeightLoss");

strcpy(ClauseVariable_List[77], "Symptom");
strcpy(ClauseVariable_List[78], "Jaundice");
strcpy(ClauseVariable_List[79], "Cramps");

strcpy(ClauseVariable_List[81], "Nausea");
strcpy(ClauseVariable_List[82], "Coughing");
strcpy(ClauseVariable_List[83], "WeightLoss");
//   strcpy(ClauseVariable_List[84], "WeightLoss");

strcpy(ClauseVariable_List[85], "Symptom");
strcpy(ClauseVariable_List[86], "Jaundice");
strcpy(ClauseVariable_List[87], "Coughing");

strcpy(ClauseVariable_List[89], "Symptom");
strcpy(ClauseVariable_List[90], "Nausea");
```

```
strcpy(ClauseVariable_List[91], "Cramps");

strcpy(ClauseVariable_List[93], "Symptom");
strcpy(ClauseVariable_List[94], "Bloating");
strcpy(ClauseVariable_List[95], "Cramps");

strcpy(ClauseVariable_List[97], "Nausea");
strcpy(ClauseVariable_List[98], "Cramps");
strcpy(ClauseVariable_List[99], "Bloating");
//   strcpy(ClauseVariable_List[100], "Bloating");









}
void Display()
{



cout<<"***** VARIABLE LIST *****\n"<<endl;

for(int z=1; z<16; z++)
```

```
cout<<"VARIABLE  "<<z<<"is --
>>"<<Variable_List[z]<<endl;

cout<<"\n";
cout<<"\n";

cout<<"PRESS ENTER TO CONTINUE"<<endl;
getchar();

/****prinint Clause Variable list **/
for(int x=1; x<28; x++)
{
cout<<"** CLAUSE **"<<x<<"\n"<<endl;
for(int y=1; y<5; y++)
{
k = 4 * (x-1) + y;
cout<<"VARIABLE   "<<y<<" --
>>"<<ClauseVariable_List[k]<<endl;
}

cout<<"\n";
if(x==8 || x==16 || x==24)
{
cout<<"PRESS ENTER TO CONTINUE"<<endl;
getchar();
}
}
}

void unconditional()
{
cout<<"\n\n ******************** PATIENT IS NOT HAVING
ANY PROBLEM   **********************\n\n"<<endl;
//exit (EXIT_FAILURE);
```

```
}
void determine_member_concl_list(void)
{

/* routine to determine if a variable (Variable) is a
member of the conclusion list (Conclusion_List).  if
yes return sn != 0. if not a member sn=0; */
/* initially set to not a member */
sn = 0;
/* member of conclusion list to be searched is f */
i = f;
while((strcmp(Variable, Conclusion_List[i]) != 0) &&
(i<60))
{/* test for membership */

i=i+1;
}
if (strcmp(Variable, Conclusion_List[i]) == 0) sn = i;




}

void push_on_stack(void)
{

/* routine to push statement number (sn) and a clause
number of 1 onto the
conclusion stack which consists of the statement stack
(statsk) and the
clause stack (clausk)..to push decrement stack pointer
(sp) */
{
```

```
//cout<<"statement number in push  is"<<sn<<"sp
is"<<sp<<endl;
sp=sp-1;

statsk[sp] = sn;

clausk[sp] = 1;

return;

}
}

void instantiate(void)
{

i=1;
/* find variable in the list */
while((strcmp(Variable, Variable_List[i]) != 0) &&
(i<60)) i=i+1;

if((strcmp(Variable, Variable_List[i]) == 0) &&
(instlt[i] != 1))
/*found variable and not already instantiated */
{
instlt[i]=1; /*mark instantiated */

switch (i)

{
```

```
case 1: cout<<"As the Variable SYMPTOM is  not
instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS ? (YES OR
NO)"<<endl;
cin>>Symptom;
if(strcmp(Symptom,"NO")==0)
unconditional();
cout<<"Answer entered by user is -->> "<<Symptom<<endl;
break;



case 2:
cout<<"As the Variable NAUSEA is not  instantiated
asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF  NAUSEA ? (YES OR
NO)"<<endl;
cin>>Nausea;
cout<<"Answer entered by user is -->> "<<Nausea<<endl;

break;

case 3: cout<<"As the Variable HEADACHE is not
instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF HEADACHE ? (YES
OR NO)"<<endl;

cin>>Headache;
cout<<"Answer entered by user is -->>
"<<Headache<<endl;
break;

case 4: cout<<"As the Variable SEIZURES is not
instantiated asking user for input \n"<<endl;
```

```
cout<<"ARE YOU HAVING ANY SYMPTOMS OF SEIZURES ? (YES
OR NO)"<<endl;

cin>>Seizures;
cout<<"Answer entered by user is -->>
"<<Seizures<<endl;
break;
case 5: cout<<"As the Variable SWELLING BRAIN is  not
instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF SWELLING BRAIN ?
(YES OR NO)"<<endl;
cin>>SwellingBrain;
cout<<"Answer entered by user is -->>
"<<SwellingBrain<<endl;
break;
case 6: cout<<"As the Variable BURNING SENSATION WHILE
URINATING is not  instantiated asking user for input
\n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF BURNING SENSATION
WHILE URINATING  ? (YES OR NO)"<<endl;
cin>>BurningSensation;
cout<<"Answer entered by user is -->>
"<<BurningSensation<<endl;
break;
case 7: cout<<"As the Variable URINE COLOR CHANGE is
not instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF  URINE COLOR
CHANGE ? (YES OR NO)"<<endl;
cin>>UrineColorChange;
cout<<"Answer entered by user is -->>
"<<UrineColorChange<<endl;
break;
case 8: cout<<"As the Variable FREQUENT URINATION is
not  instantiated asking user for input \n"<<endl;
```

```
cout<<"ARE YOU HAVING ANY SYMPTOMS OF FREQUENT
URINATION ? (YES OR NO)"<<endl;
cin>>FrequentUrination;
cout<<"Answer entered by user is -->>
"<<FrequentUrination<<endl;
break;

case 9: cout<<"As the Variable VOMIT is not
instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF VOMIT ? (YES OR
NO)"<<endl;
cin>>Vomit;
cout<<"Answer entered by user is -->> "<<Vomit<<endl;
break;

case 10:
cout<<"As the Variable COUGHING is not  instantiated
asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF COUGHING ? (YES
OR NO)"<<endl;
cin>>Coughing;
cout<<"Answer entered by user is -->>
"<<Coughing<<endl;
break;


case 11: cout<<"As the Variable WHEEZING is not
instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF  WHEEZING ? (YES
OR NO)"<<endl;
cin>>Wheezing;
cout<<"Answer entered by user is -->>
"<<Wheezing<<endl;
break;
```

```
case 12: cout<<"As the Variable WEIGHT LOSS is not
instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF WEIGHT LOSS? (YES
OR NO)"<<endl;
cin>>WeightLoss;
cout<<"Answer entered by user is -->>
"<<WeightLoss<<endl;
break;

case 13: cout<<"As the Variable CRAMPS is not
instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF  CRAMPS ?(YES OR
NO)"<<endl;
cin>>Cramps;
cout<<"Answer entered by user is -->> "<<Cramps<<endl;
break;

case 14: cout<<"As the Variable JAUNDICE is  not
instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF  JAUNDICE ? (YES
OR NO)"<<endl;
cin>>Jaundice;
cout<<"Answer entered by user is -->>
"<<Jaundice<<endl;
break;
case 15: cout<<"As the Variable BLOATING is not
instantiated asking user for input \n"<<endl;
cout<<"ARE YOU HAVING ANY SYMPTOMS OF  BLOATING ? (YES
OR NO)"<<endl;
cin>>Bloating;
cout<<"Answer entered by user is -->>
"<<Bloating<<endl;
break;
```

```
/* end of inputs statements for sample position
knowledge base */

}

}
}


#include<iostream>
#include <stdio.h>
#include <string.h>
#include "forward.h"
//#include<windows.h>

using namespace std;


int main()
{

            initialize();
            display();
   do{
        initialize();
         cout<<"\n";

cout<<"==========================================
==================";
         cout<<"\n\n\n\n\n";
         cout<<"                WELCOME TO Cancer
TREATMENT CLINIC     ";
         cout<<"\n\n\n\n\n";
```

```
cout<<"================================================
==================\n\n";

        cout<<"\n The following Cancers are treated in
this clinic : \n";
        cout<<" 1. Brain Cancer \n";
        cout<<" 2. Blader Cancer \n";
        cout<<" 3. Lung Cancer \n";
        cout<<" 4. Pancreatic Cancer \n";
        cout<<" 5. Ovarian Cancer \n";




    /****** INFERENCE SECTION *****************/


     strcpy(c,"cancer");

    strcpy(cndvar[bp], c);

    bp = bp + 1;
    /* set the condition variable pointer consisting of
the
    statement number (sn) and the clause number (cn) */
    sn = 1; cn = 1;
    /* find the next statement number containing the
condition variable
    which is in front of the queue (cndvar), this
statement number
    is located in the clause variable list
(ClauseVariable_List) */
```

```
    /* start at the beginning */
    f=1;
    b496: search();
    /* point to first clause in statement */
    cn=1;
    if (sn != 0)
        /* more statements */
    {

        /* locate the clause */
        i = 4 * (sn-1) + cn;
        /* clause variable */
        strcpy(v, ClauseVariable_List[i]);
        /* are there any more clauses for this
statement */
        while (strcmp(v, "") !=0)
            /* more clauses */
        {
            /* check instantiation of this clause */
            check_instantiation();
            cn = cn+1;
            /* check next clause */
            i = 4 * (sn-1) + cn;
            strcpy(v, ClauseVariable_List[i]);
        }

        /* no more clauses - check IF part of statement
*/
        s = 0;


        /* sample IF-THEN statements from the position
knowledge base */
        switch(sn)
```

```
        {
        case 1: if (strcmp(cancer, "BrainCancer") == 0)
s=1;
            break;
        case 2: if ( strcmp(cancer, "BladderCancer") ==
0) s=1;
            break;
        case 3: if (strcmp(cancer, "LungCancer") ==0)
s=1;
            break;
        case 4: if ( strcmp(cancer, "PancreaticCancer")
== 0) s=1;
            break;
        case 5: if ( strcmp(cancer, "OvarianCancer") ==
0) s=1;
            break;


        }




        /* see if the THEN part should be inovked,
i.e., s=1 */
        if (s != 1)
        {   //cout<<"entered the s !=1 loop\n");
            f = sn + 1;
            goto b496;
        }



        /* invoke THEN part */
```

```
        switch (sn)
        {

            /* put variable on the conclusion variable
queue */

        case 1:
            {
            strcpy(treatment, "Chemotherapy");

cout<<"\n=============================================
====================\n\n\n";
            cout<<"                          TREATMENT =
Chemo Therapy\n";

cout<<"\n\n\n=============================================
========================\n\n\n";
            strcpy(v, "treatment");
            instantiate();
            break;}

        case 2:
            strcpy(treatment, "Transurethral Resuction
Of Bladder");

cout<<"\n=============================================
====================\n\n\n";
            cout<<"                TREATMENT =
Transurethral Resuction Of Bladder\n";

cout<<"\n\n\n=============================================
========================\n\n\n";
            strcpy(v, "treatment");
            instantiate();
```

```
            break;

        case 3:
            strcpy(treatment, "Radiation");

cout<<"\n================================================
====================\n\n\n";
            cout<<"              TREATMENT =
Radiation\n";

cout<<"\n\n\n================================================
=========================\n\n\n";
            strcpy(v, "treatment");
            instantiate();
            break;
        case 4:
            strcpy(treatment, "Surgery");

cout<<"\n================================================
====================\n\n\n";
            cout<<"           TREATMENT = Surgery\n";

cout<<"\n\n\n================================================
=========================\n\n\n";
            strcpy(v, "treatment");
            instantiate();
            break;
        case 5:
            strcpy(treatment, "Surgery and Chemo
Therapy");

cout<<"\n================================================
====================\n\n\n";
```

```
            cout<<"                              TREATMENT =
Surgery and Chemo Therapy\n";

cout<<"\n\n\n=========================================
=========================\n\n\n";
            strcpy(v, "treatment");
            instantiate();
            break;


         }
         f = sn + 1;
         goto b496;
       }

      /* no more clauses in the clause variable list
(ClauseVariable_List)
      containing the variable in front of the queue
(cndvar(fp))
      then remove front variable (cndvar(fp)) and
replace it by
      the next variable (cndvar(fp+1)). If no more
variables are
      at the front of the queue, stop. */
      /* next queue variable */
      fp=fp+1;
       if (fp < bp)
       {
         /* check out the condition variable */
         f = 1;
         goto b496;
       }
      /* no more conclusion variables on queue */
```

```
     cout<<"\n\n Do you want to continue? Press 'Y' for
yes and 'N' to exit.";
     cout<<"Any other character will exit. ";
     cin>>contin;
     cout<<"\n\n";
     }while(strcmp(contin,"Y") == 0  ||
strcmp(contin,"y") == 0);

    system("PAUSE");
}
```

```
#ifndef FORWARD_H_
#define FORWARD_H_
#include<iostream>
#include <stdio.h>
#include <string.h>

using namespace std;


int flag;
char cndvar[10][15];
char Variable_List[20][20], /* variable list*/
ClauseVariable_List[20][20]; /* clause var list */
char c[20], vp[15], /* condition variable */
v[15]; /*variable */
char treatment[50];
char cancer[30];
char Position[15], /* position */  qu[15]; /* qualify
*/
char size[50];
```

```
int Instantiated_List[10];          /* instantiated
list*/
int f, i, j, k, s, fp   /* front pointer */;
int  bp  /* back pointer */,  gr /* grade */,  sn; /*
statement number */
int cn;  /* clause number */
int choice;
char contin[2];

void initialize(void);
void display(void);
void search(void);
void check_instantiation(void);
void instantiate(void);


//
==========================================================
==================
/* Iinitialize the clause variable list, conclusion
variable queue and other variables
to initial values as per the forward chaining
algorithm*/

void initialize()
{


   fp=1;
    bp=1;

    for (i=1;i < 20; i++)
        strcpy(ClauseVariable_List[i], "");
    for (i=1;i < 11; i++)
```

```
        strcpy(cndvar[i], "");
    for (i=1;i < 11; i++)
        Instantiated_List[i]=0;
    for (i=1;i < 11; i++)
        strcpy(Variable_List[i], "");
    for (i=1;i < 11; i++)
    {
        strcpy(cndvar[i], "");
        strcpy(Variable_List[i], "");
        Instantiated_List[i]=0;
    }

    /* enter variables which are in the IF part, 1 at a
time in
    the exact order that they occur. Up to 3 variables
per
    IF statement. Do not duplicate any variable names.
Any
    name is used only once. If no more variables left,
just
    hit return key */
    strcpy(Variable_List[1], "cancer");


    /* enter variables as they appear in the IF
clauses, Up to 3
    variables per IF statement. If no more variables
left, just
    hit return key */

    strcpy(ClauseVariable_List[1], "cancer");
    strcpy(ClauseVariable_List[5], "cancer");
    strcpy(ClauseVariable_List[9], "cancer");
    strcpy(ClauseVariable_List[13], "cancer");
```

```
    strcpy(ClauseVariable_List[17], "cancer");

}
    /* Display the values in VARIABLE LIST and CLAUSE-
VARIABLE POINTER */
    void display(void)
    {
         cout<<"************ VARIABLE LIST
**************\n";
                 cout<<Variable_List[1]<<endl;

         cout<<"\n********** CLAUSE-VARIABLE LIST
**********\n";
         for (i = 1; i < 6; i++)
             {

                 for (j = 1; j < 5; j++)
                 {
                     k = 4 * (i - 1) + j;
                     cout<<"VARIABLE   "<<j<<" --
>>"<<ClauseVariable_List[k]<<endl;
                 }

                 }



    }



    //
==========================================================
==================
```

```
    /* Search clause variable list for a varialbe
(ClauseVariable_List) equal to the
    one in front of the conclusion queue (cndvar).
Return the statement
    number (sn). If there is no match, i.e., sn=0, the
first statement
    for the space is f. */
    void search()
    {

        flag = 0;
        sn = f;

        while ((flag == 0) && (sn <= 19))
        {
            cn=1;
            k = (sn-1)*4+cn;
            while ((strcmp(ClauseVariable_List[k],
cndvar[fp]) != 0) && (cn < 5))
            {
                cn = cn+1;
                k = (sn-1)*4+cn;

            }

            if (strcmp(ClauseVariable_List[k],
cndvar[fp]) == 0) flag = 1;
            if (flag == 0) sn = sn+1;
        }
        if (flag == 0) sn=0;
    }
```

```
    //
==========================================================
==================
    /* Routine to instantiate a variable (v) if it
isn't already.
    The instantiate indication (Instantiated_List) is a
0 if not, a 1 if it is.
    The vriable list (Variable_List) contains the
variable (v) */
    void check_instantiation()
    {
        i=1;

        /* find variable in the variable list */
        while ((strcmp(v, Variable_List[i]) != 0) && (i
< 4))
         i = i+1;

        /* check if already instantiated */
        if (Instantiated_List[i] != 1)
        {
            /* mark instantiated */
            Instantiated_List[i] = 1;
            /* the designer of this knowledge base
places the input
            statements to instantiate the variables in
this case
            statement */

            cout<<"\nPlease enter the number
corresponding to the cancer for which treatment is
required : ";
```

```
            cin>>choice;

                    if( choice == 1)
                    {
                   cout<<"\nRule 1 is executed \n\n";
                   cout<<"Brain Cancer variable is
instantiated\n\n";
                      strcpy(cancer,"BrainCancer");
                    }
                    else if(choice == 2)
                    {
                     cout<<"\nRule 2 is executed \n\n";
                       cout<<"Bladder Cancer variable is
instantiated\n\n";
                      strcpy(cancer,"BladderCancer");
                    }
                    else if(choice == 3)
                    {
                     cout<<"\nRule 3 is executed \n\n";
                       cout<<"Lung Cancer variable is
instantiated\n\n";
                      strcpy(cancer,"LungCancer");
                    }
                    else if(choice == 4)
                    {
                     cout<<"\nRule 4 is executed \n\n";
                       cout<<"Pancreatic Cancer variable
is instantiated\n\n";

strcpy(cancer,"PancreaticCancer");
                    }
                    else if(choice == 5)
                    {
                     cout<<"\nRule 5 is executed \n\n";
```

```
                        cout<<"Ovarian Cancer variable is
instantiated\n\n";
                       strcpy(cancer,"Ovarian Cancer");
                    }

                    else
                       {
                       cout<<"\n ENTER NUMBERS FROM 1-18
ONLY \n\n";
                      }

                      /* end of input statements for the
position knowledge base */
    }


    }


    //
==========================================================
==================
    /* Routine to instantiate a varialbe (v) and then
place it on the
    back of the queu (cndvar[bp]), if it is not already
there. */
    void instantiate()
    {
        i=1;
        /* find varialbe in the varialbe list
(Variable_List) */
        while ((strcmp(v, Variable_List[i]) != 0) && (i
<= 10)) i=i+1;
```

```
        /* instantiate it */
        Instantiated_List[i] = 1;
        i = 1;

        /* determine if (v) is or already has been on
the queue (cndvar) */
        while ((strcmp(v, cndvar[i]) != 0) && (i <=
10)) i=i+1;
        /* variable has not been on the queue. Store it
in the back of the queue */
        if (strcmp(v, cndvar[i]) != 0)
        {
            strcpy(cndvar[bp], v);
            bp=bp+1;
        }

    }




#endif /* FORWARD_H_ */
```

17. MODIFICATIONS TO THE PROGRAM:

The code provided by Dr.Moonis Ali is erroneous and less efficient. So, we made the following changes to the coding.

o The entire program is rewritten in C++, whereas the original program was in C. o Removed all syntax errors.

o Unused variables are removed from the program, thereby improved the memory efficiency.

o All declarations were made in a separate header file and then included in the program, so as to improve easy understanding.

o All the variables were given meaningful names in such a way that they can be easily understood. o Included a function display(), to display all the data structures used in the program.

o Most of the character arrays are converted to String arrays so that they can be easily handled in C++.

o The given program did not have few necessary library files and I have included them namely,

#include<iostream>, #include<string> and also included 'using namespace std;'.

18. CONCLUSION:

The project is successfully designed and implemented using Forward and Backward chaining algorithms to obtain optimal solution.