THE POONA GUJRATHI KELVANI MANDAL'S

Haribhai V. Desai College

Department of Computer Science

2020-2021

A Project Report on

C TUTOR

BY

Durvesh Ramane (B-3936)

Under guidance of

Mrs Jyoti Malusare

BACHELORS OF COMPUTER SCIENCE

Savitribai Phule Pune university

Academic year 2020-2021

Department of computer science

ACKNOWLEDGEMENT

I am grateful to be given opportunity to present this project as a part of the curriculum of Bsc. Computer Science. I would like to take this opportunity to appreciate the constructive inputs, encouragement and help of all staff member and fellow colleagues in the development of our project.

I would like to express my deepest appreciation to my project guide Mrs. Jyoti Malusare for her immense support , guidance and assistance throughout the course of my project.

I also thank all those who have directly or indirectly guided and helped me in completion of this project.

Thanking you all

Durvesh Ramane

Index

1. INTRODUCTION

1.1 Detailed Problem Definition

The current learning system is very limited to few resources, students are unable to get knowledge more than that the lecture provides to them. This is the end limit of the student's performances, because everything a student gets is collected from lectures in class.

Here are some of the problems of the current system:

- For new lectures to a course students have to get materials on their own.
- Students are required to be present in the classroom in order to gain knowledge there by sacrificing all other responsibilities.
- Students are unable to share resources effectively and hold group discussions that are monitored or supervised by lectures.
- Some students often find it difficult to understand and prefer more visual methods of learning through videos and tutorials.
- For some students it is not possible to attend each and every lecture and this end up missing out the content delivered in the classes.

1.2 Presently Available Systems for the same

Currently available system is confirmed to the classroom with lectures delivery as the only method of conveying knowledge rather than an electronic means of learning. The current learning system is very limited to few resources, students are unable to get knowledge more than that the lecture provides to them.

For some students it is not possible to attend each and every lecture and this end up missing out the content delivered in the classes. Some students often find it difficult to understand and prefer more visual methods of learning through videos and tutorials. So, it is required to develop an learning application that will overcome all these problems and help students to learn while they are not in the class or lectures.

1.3 Need for the New System

Currently we have many websites (e.g.: www.tutorialspoints.com , www.geeksforgeeks.com , etc.) which provide sort of readymade codes and all information related to that but the biggest limitation is that these websites are not user friendly and user have to became paid member to access entire study material which is also not efficient at times.

So it is required to develop an learning application which will provide all necessary information related to a particular topic and everyone can access it without any difficulty and it should be free for all those students who cannot purchase membership of these websites (e.g.: www.tutorialspoints.com , www.geeksforgeeks.com , etc.)

1.4 Scope of the Proposed System

The main objective of the project is to manage the details of the topics and to provide demo related to that topic. In this application , there is no restriction to the user and he/she can easily access all materials related to any topic and can see demo. Students can rewatch the tutorial if they are not able to understand any part of it. The navigation will be smooth and student can study whichever topic they wish to at any point.

The purpose of the project is to build an application program to reduce the manual work for managing the different topics related to a particular course. It will also reduced the cost of collecting the study material.

2. ANALYSIS

2.1 FEASIBILITY STUDY

Once the problem is clearly understood, the next step is to contact feasibility study, which is high-level capsule version of the entered systems and design process. The objective is to determine whether or not the proposed system is feasible. Hence, we consider the three aspects of feasibility.

2.1.1 Technical Feasibility

It analyses and determines whether the solution can be supported by existing technology or not. The analyst determines whether current technical resources are to be upgraded or added to fulfill the new requirements.

The System available and hardware components used are:

- Fedora 33 , Kernel Linux , GNOME 3.38.5
- Hardware memory - 3.7 GB
- System status – Available disk space: 856.9 GB
- JAVA

2.1.2 Economical Feasibility

It is evaluating the effectiveness of candidate system by using cost/benefit analysis method. Its main aim is to estimate the economic requirements of candidate system before investing funds are committed to proposal.

The system is economically feasible. Most of the requirements are readily available. The operating system be used that is Cent OS is free, the java application can be installed, all the software requirements won't require any extra charges. We'll have to pay for the internet connection if required and the hardware.

2.1.3 Operational Feasibility

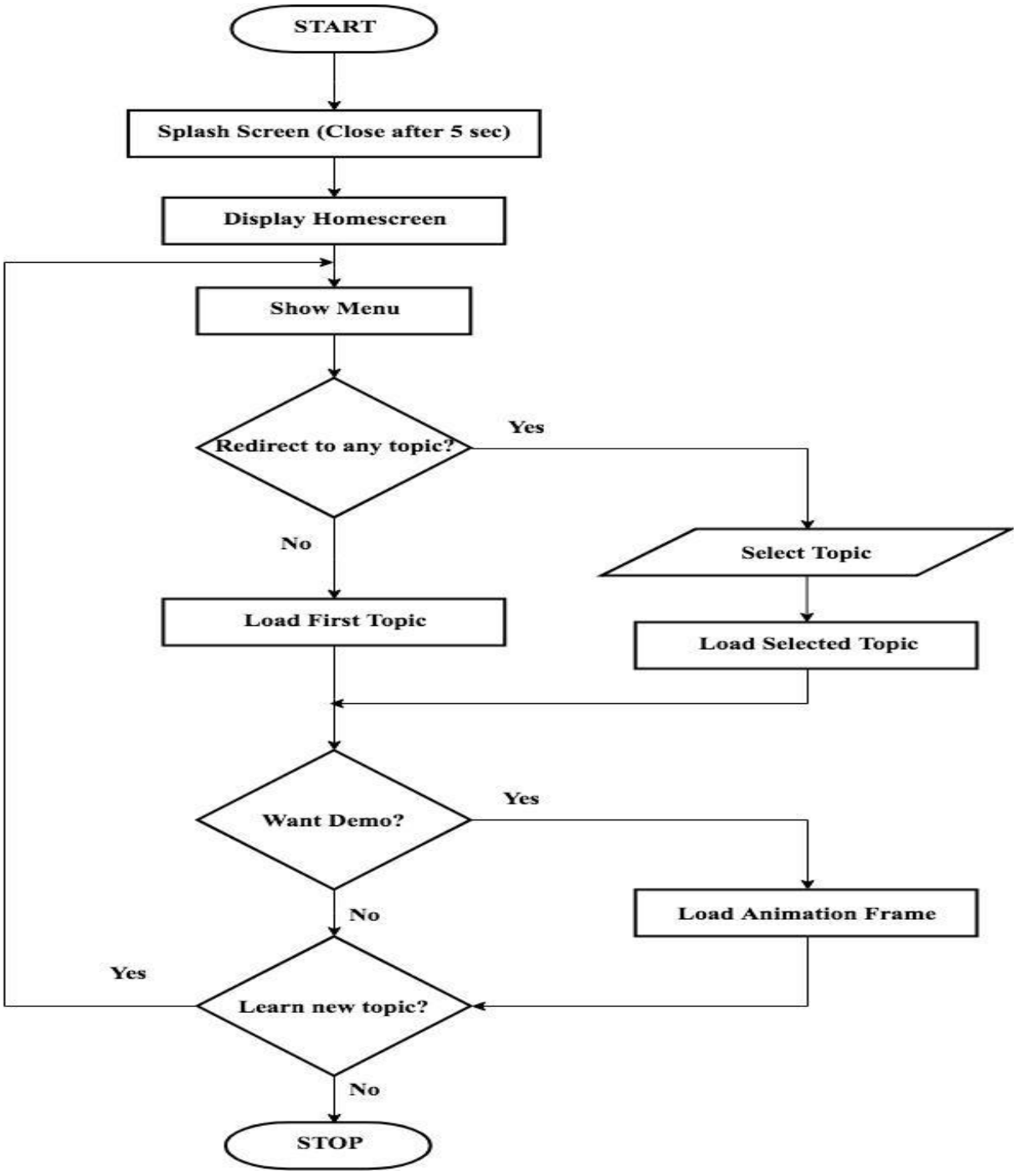
No doubt the proposed system is fully GUI based that is very user friendly and all inputs to be taken are all self-explanatory even to a layman. Besides, a proper training has been conducted to let know the essence of the system to the users so that they feel comfortable with new system. As far our study is concerned the clients are comfortable and happy as the system has cut down their loads and doing.

In this case the operationally implementation cost of the system considered after implementing the system training programs are changed or arranged for user. Cost of training programs and space requirement is also considered.

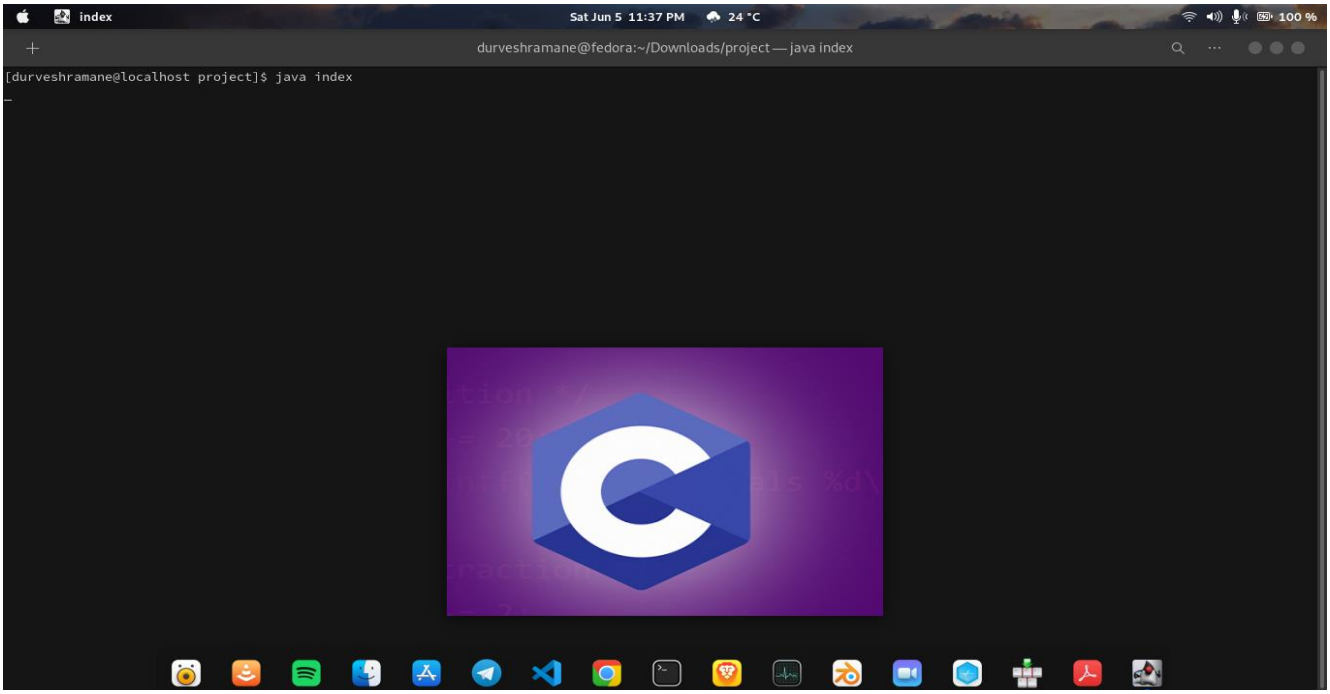
Our developers, experts coordinate with staff and give the information about the operation/working of the current system. This system is easy to work with.

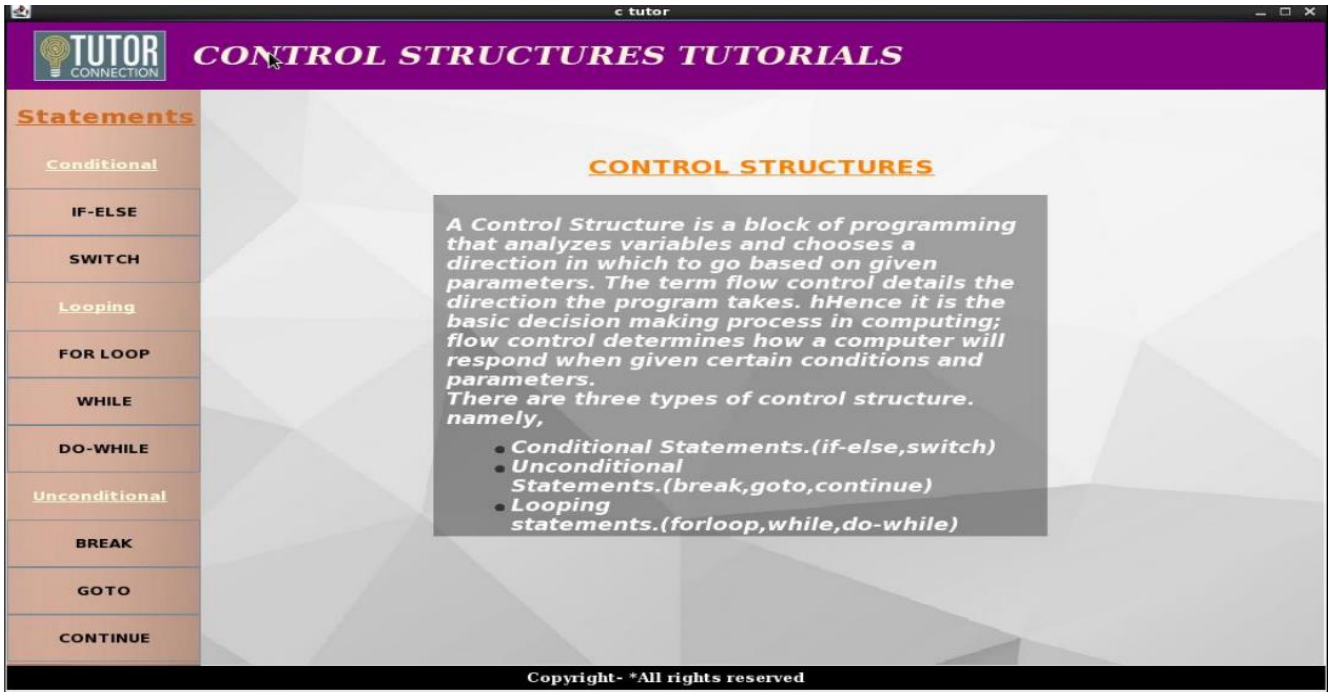
3. Design

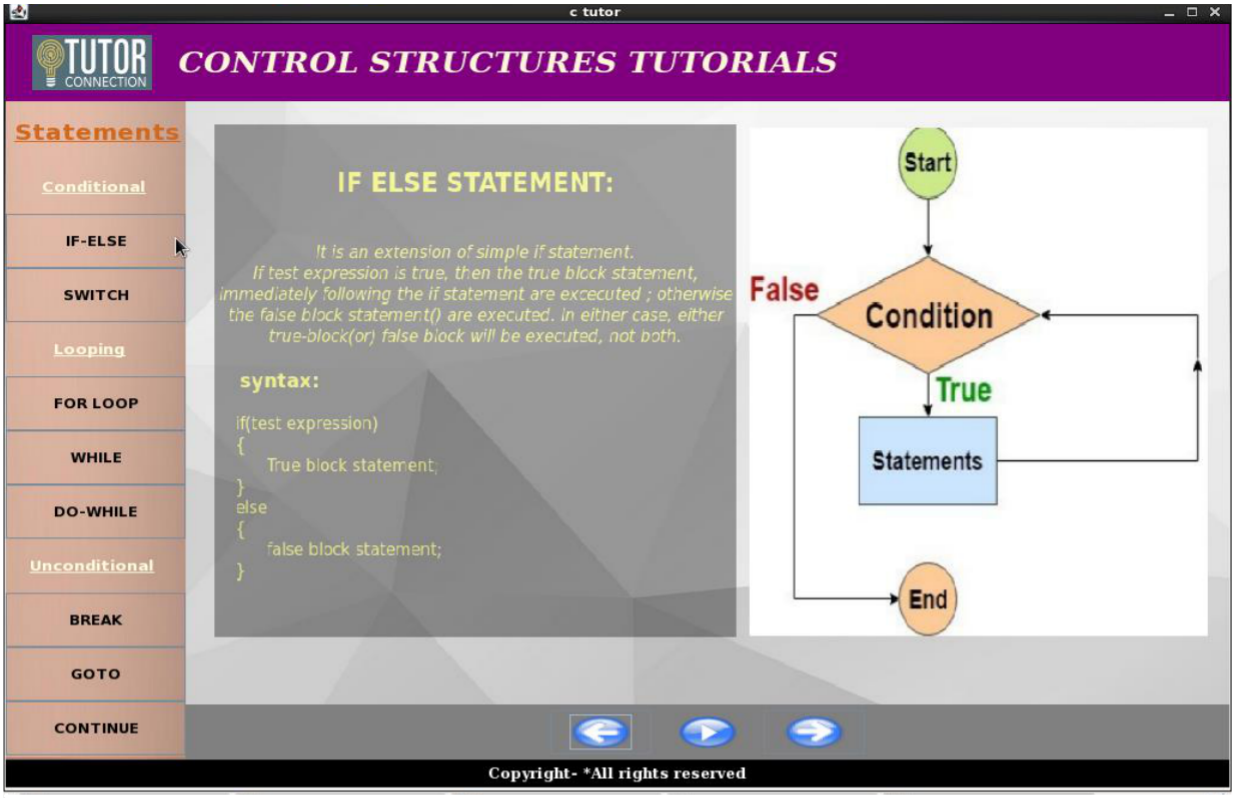
3.1 Flowchart

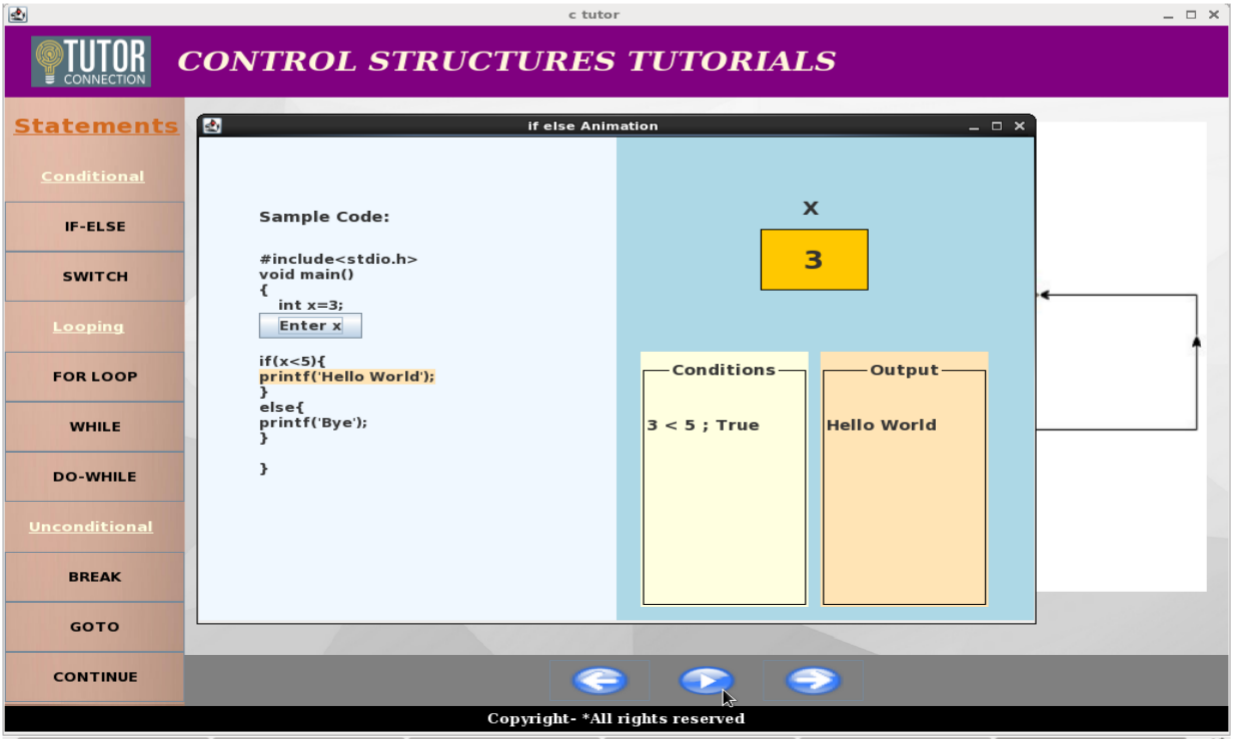


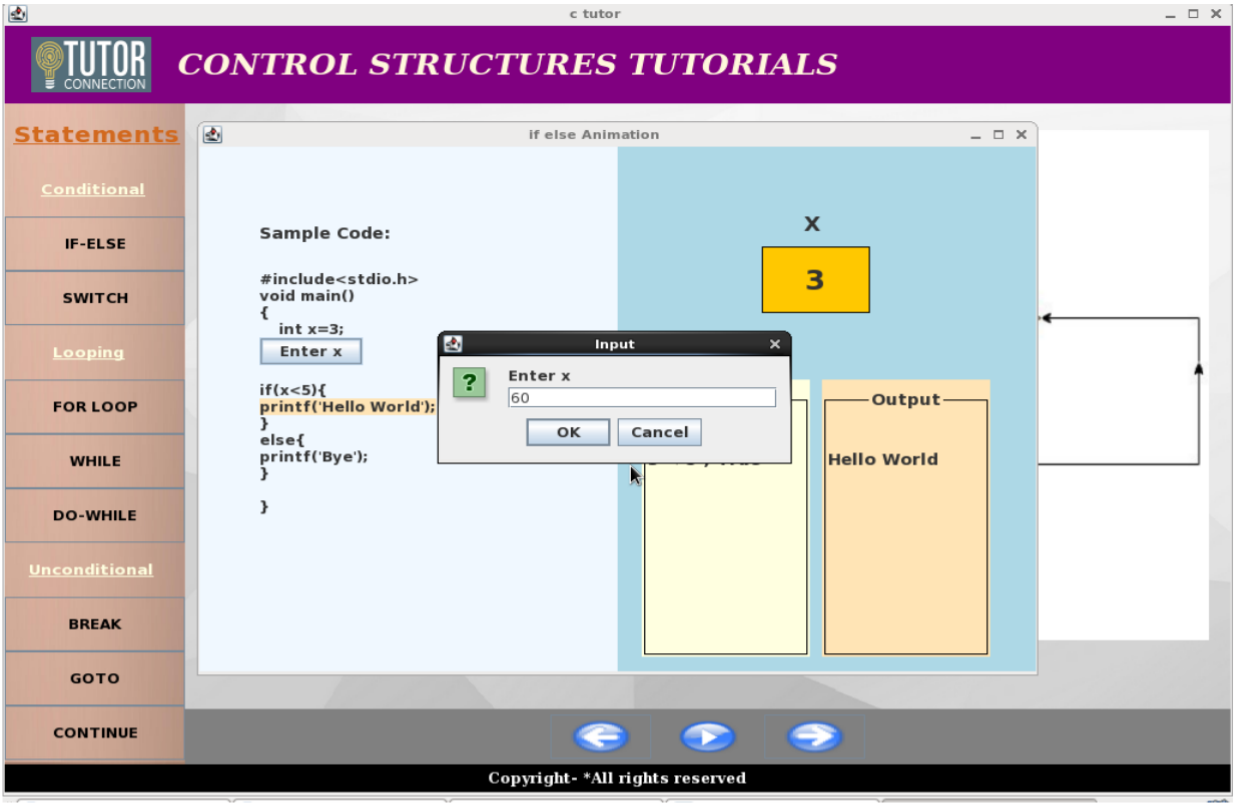
3.2 Input and Output Screens and Reports

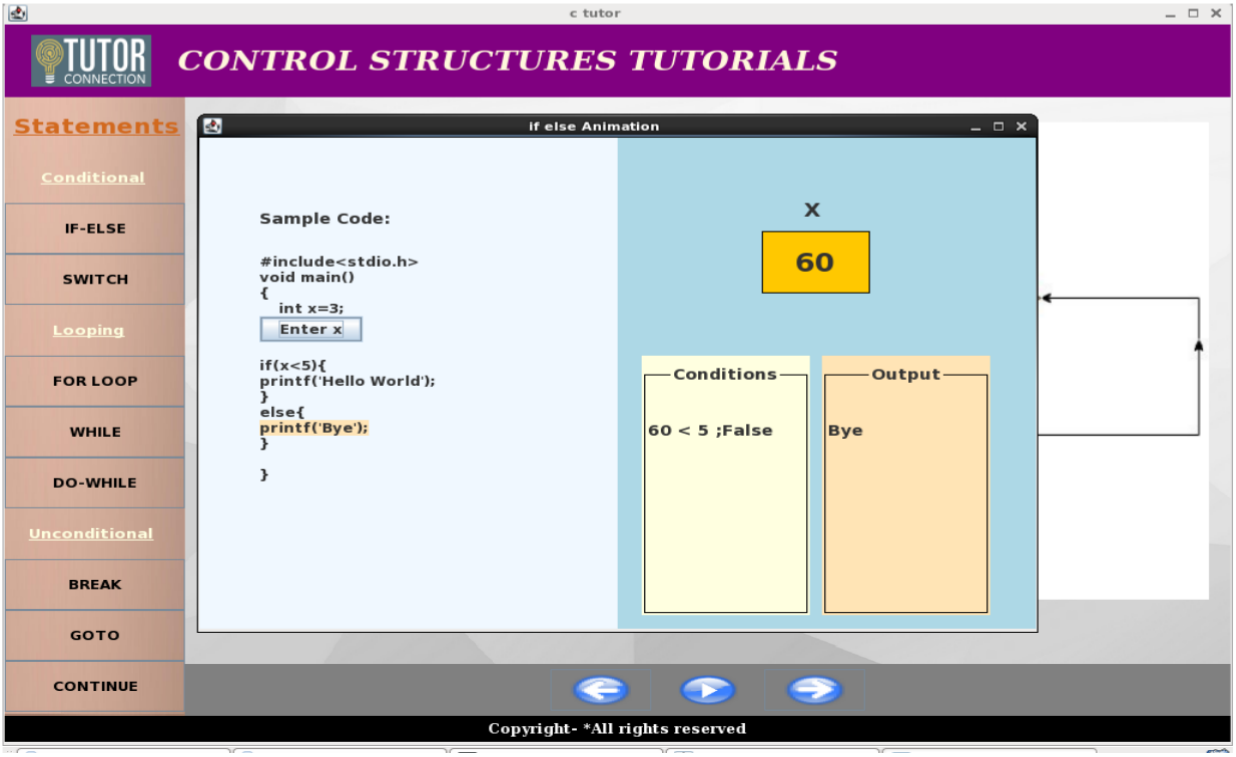


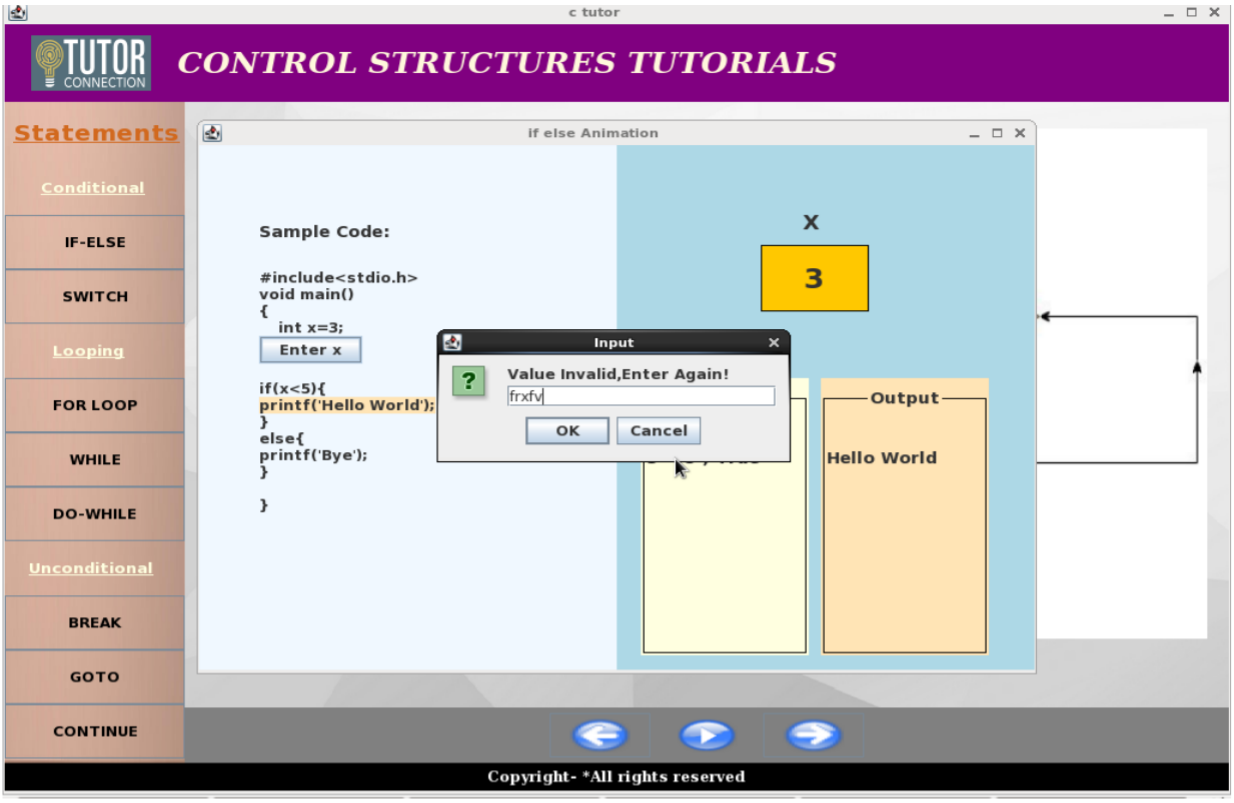


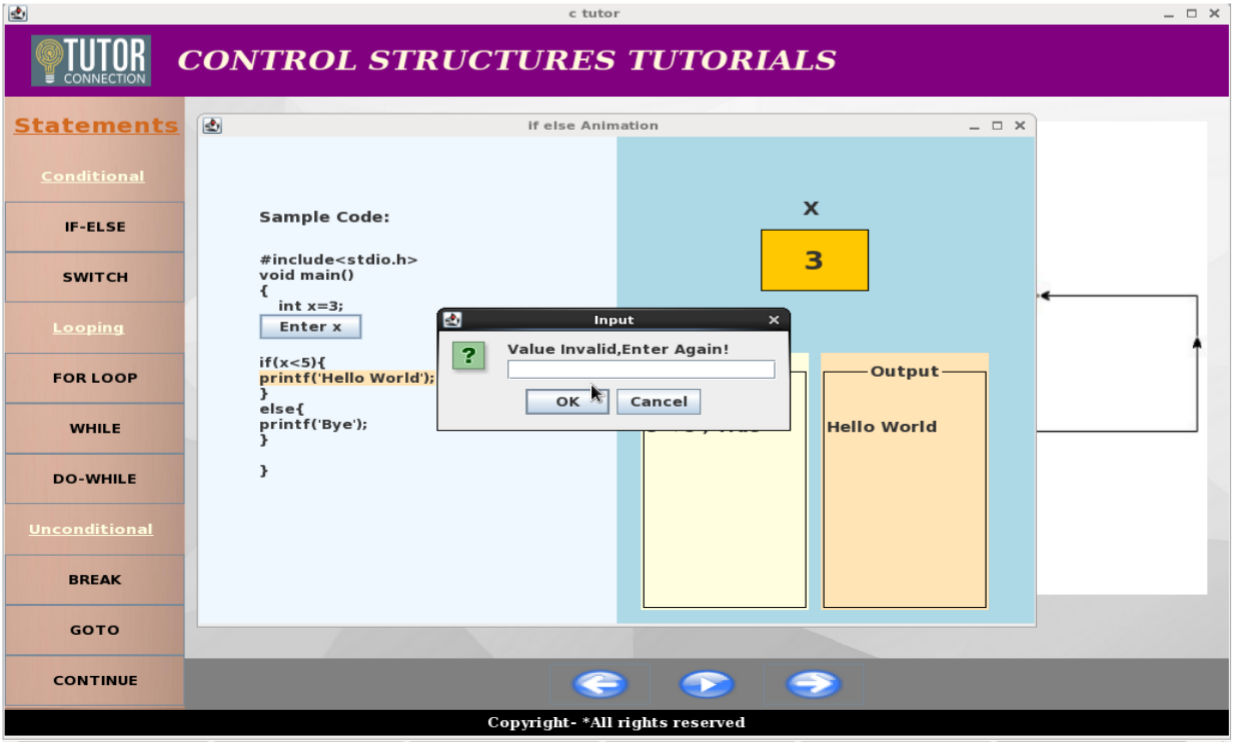

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
IF ELSE STATEMENT:
It is an extension of simple if statement.
If test expression is true, then the true block statement,
immediately following the if statement are excecuted ; otherwise
the false block statement() are executed. in either case, either
true-block(or) false block will be executed, not both.
syntax:
if(test expression)
{
True block statement;
}
else
{
false block statement;
}
Start
False
Condition
True
Statements
End
Copyright- *All rights reserved

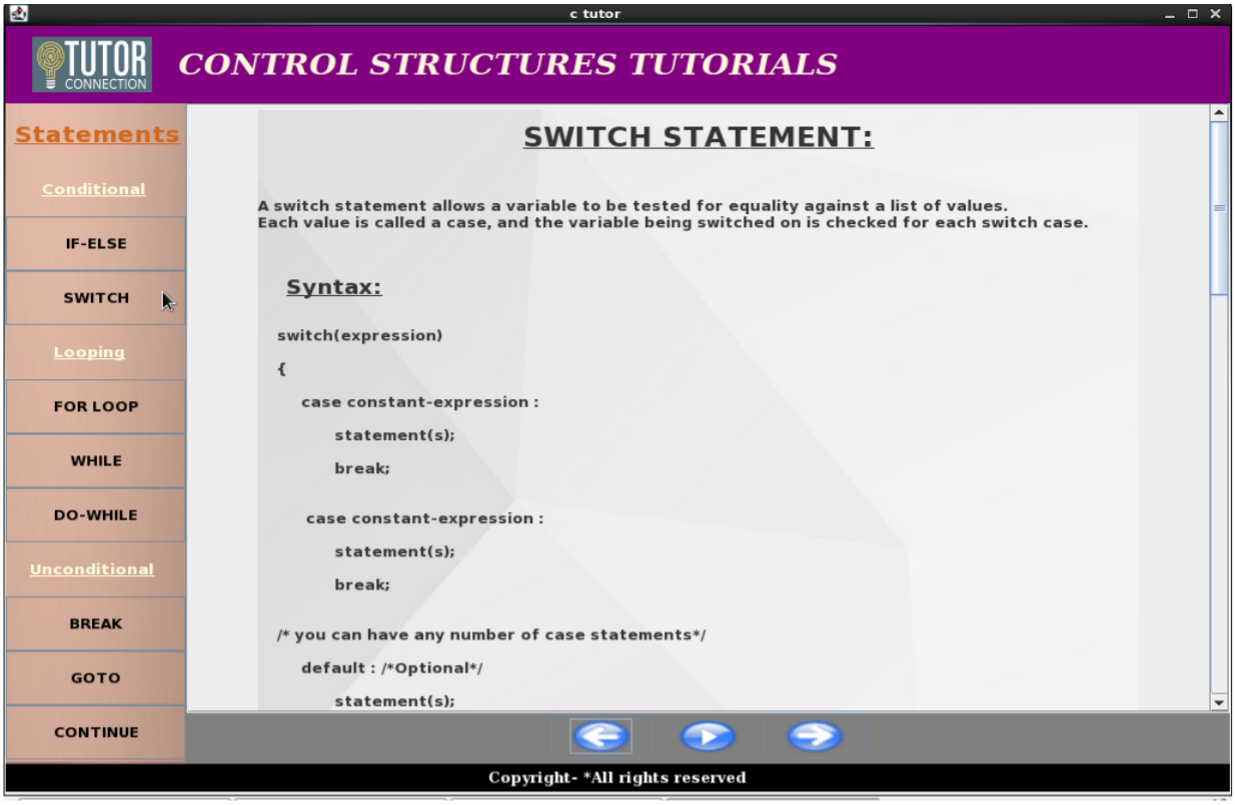

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
if else Animation
Sample Code:
#include<stdio.h>
void main()
{
int x=3;
Enter x
if(x<5){
printf('Hello World');
}
else{
printf('Bye');
}
}
x
3
Conditions
3 < 5 ; True
Output
Hello World
Copyright- *All rights reserved

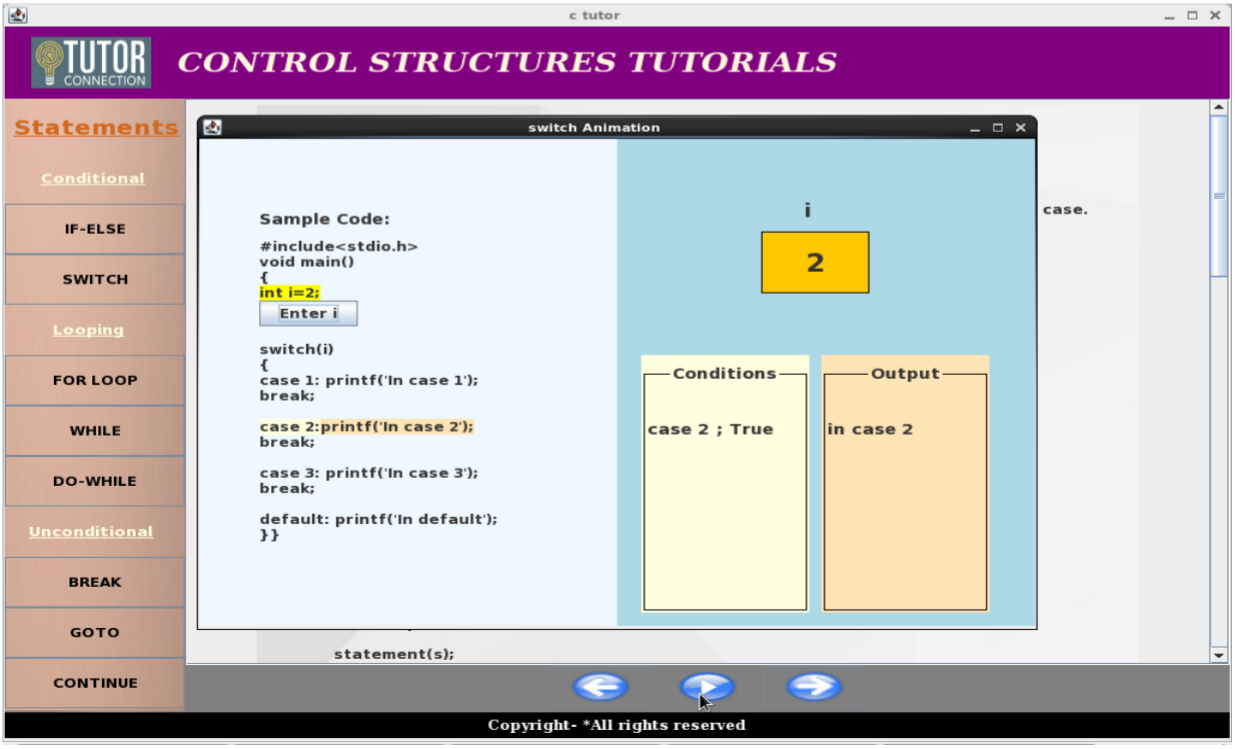

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
if else Animation
Sample Code:
#include<stdio.h>
void main()
{
int x=3;
Enter x
if(x<5){
printf('Hello World');
}
else{
printf('Bye');
}
}
X
3
Input
Enter x
60
OK
Cancel
Output
Hello World
Copyright- *All rights reserved

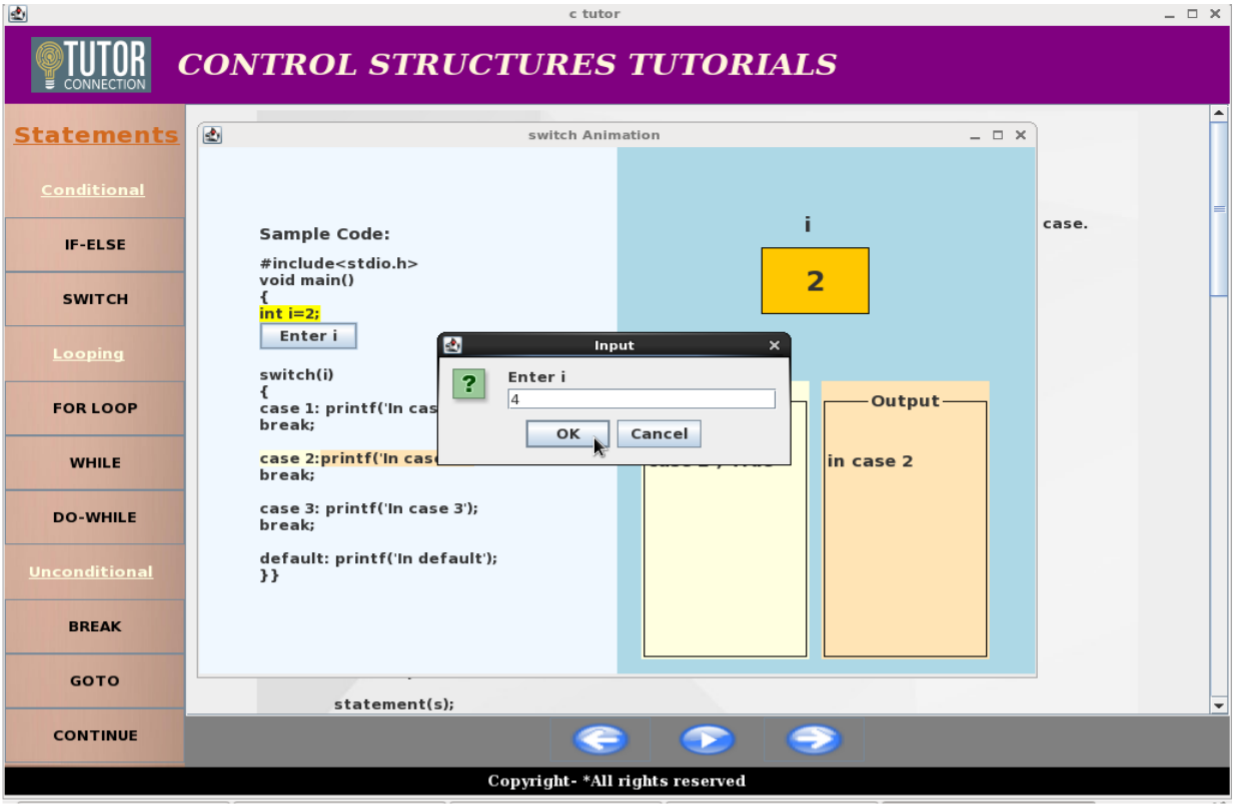

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
if else Animation
Sample Code:
#include<stdio.h>
void main()
{
int x=3;
Enter x
if(x<5){
printf('Hello World');
}
else{
printf('Bye');
}
}
X
60
Conditions
60 < 5 ;False
Output
Bye
Copyright- *All rights reserved

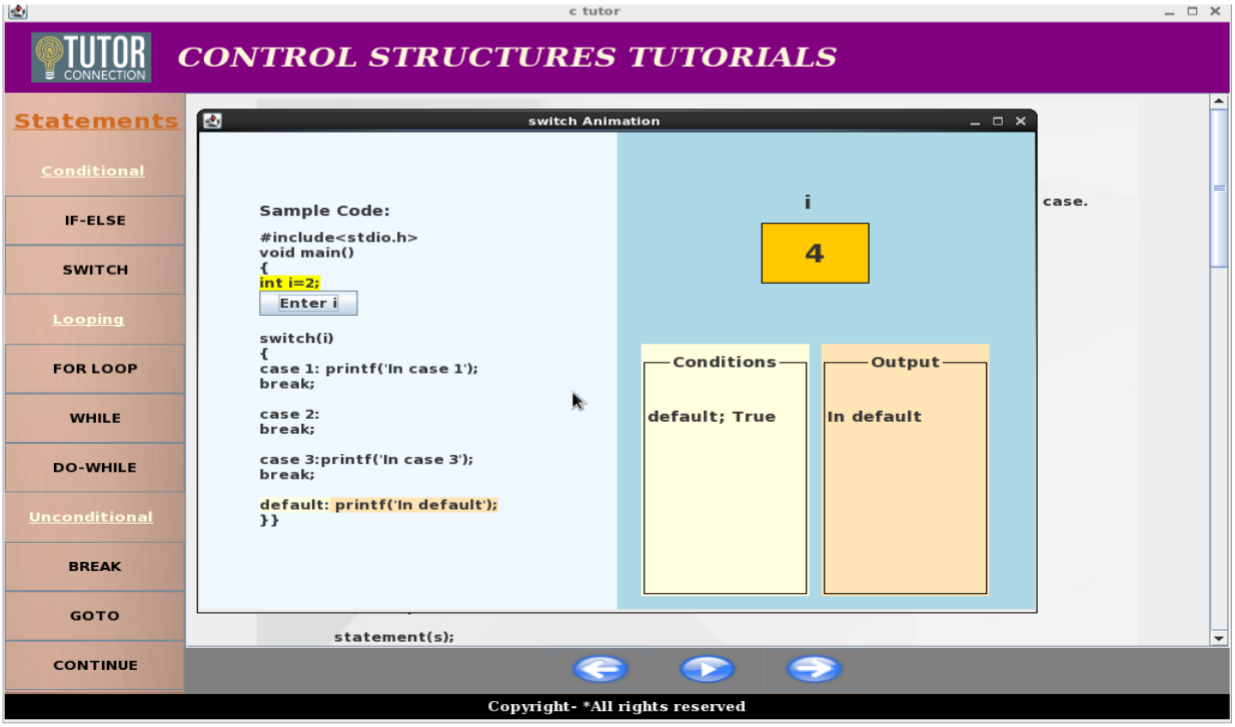

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
if else Animation
Sample Code:
#include<stdio.h>
void main()
{
int x=3;
Enter x
if(x<5){
printf('Hello World');
}
else{
printf('Bye');
}
}
X
3
Input
Value Invalid,Enter Again!
frxfv
OK
Cancel
Output
Hello World
Copyright- *All rights reserved

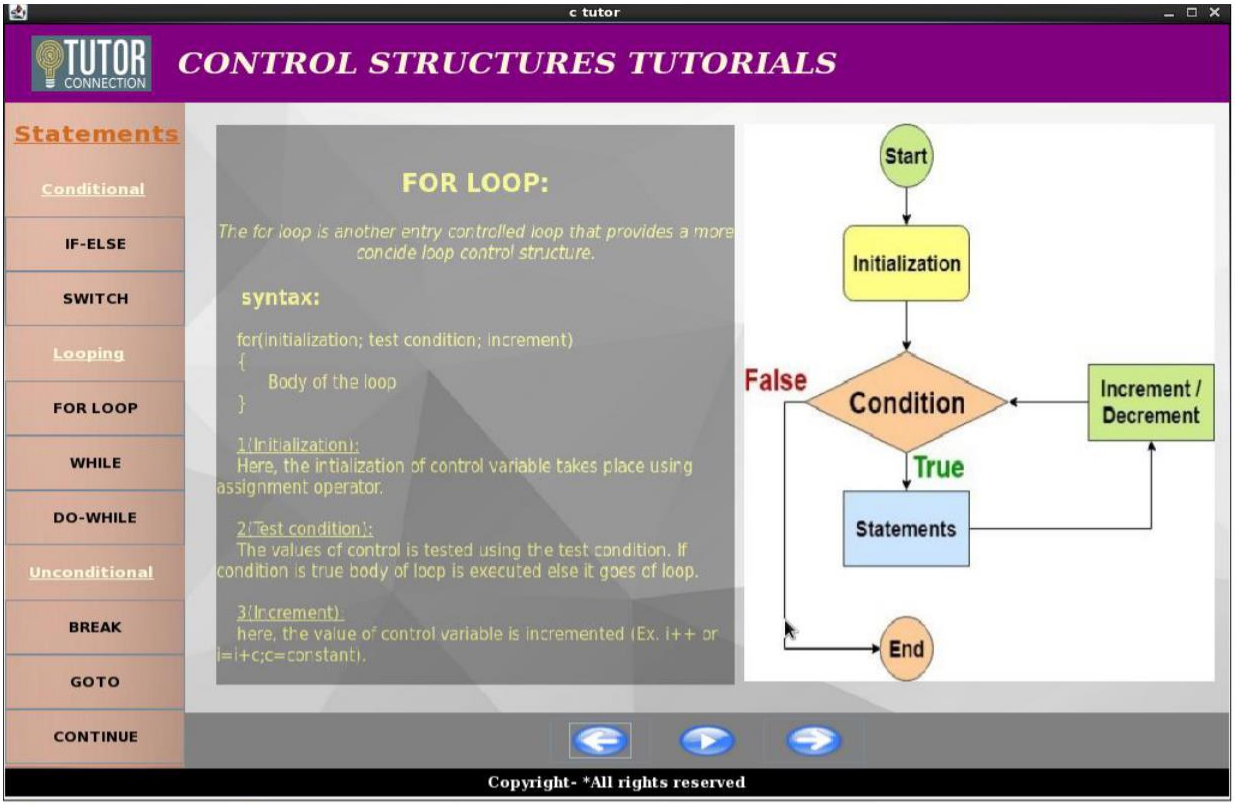

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
if else Animation
Sample Code:
#include<stdio.h>
void main()
{
int x=3;
Enter x
if(x<5){
printf('Hello World');
}
else{
printf('Bye');
}
}
X
3
Input
Value Invalid,Enter Again!
OK
Cancel
Output
Hello World
Copyright- *All rights reserved

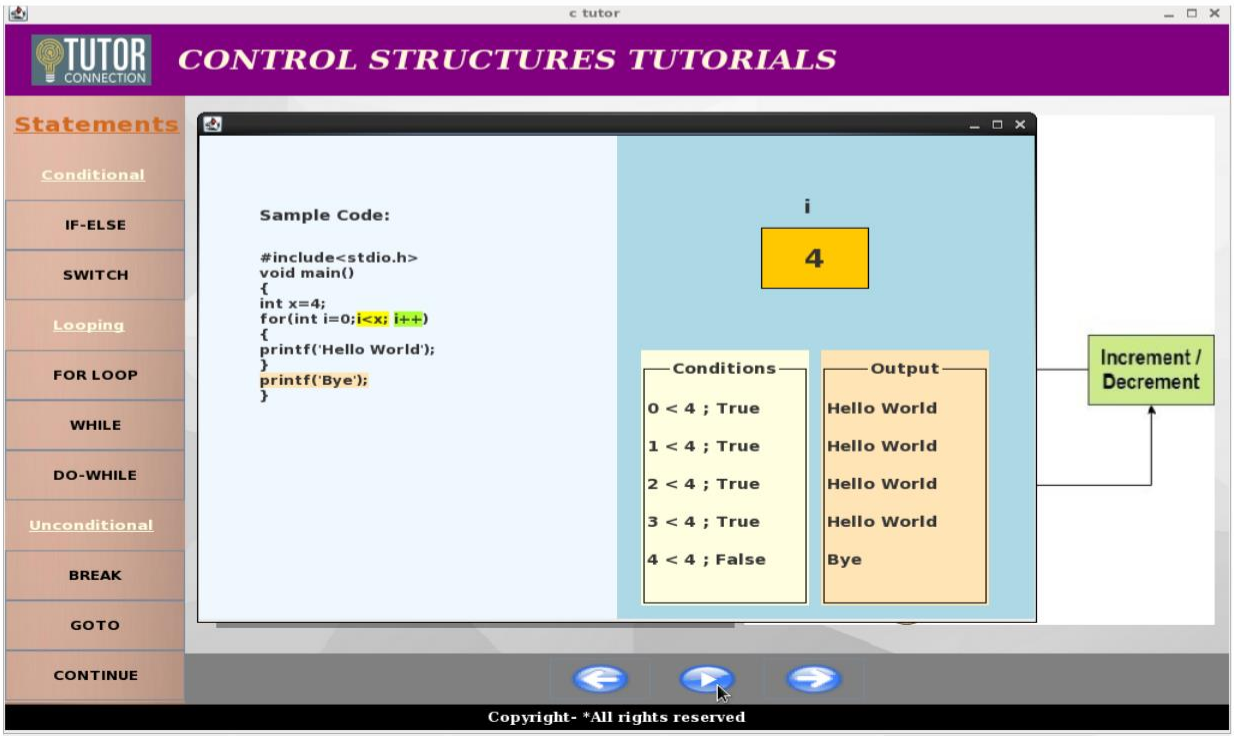

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
SWITCH STATEMENT:
A switch statement allows a variable to be tested for equality against a list of values.
Each value is called a case, and the variable being switched on is checked for each switch case.
Syntax:
switch(expression)
{
case constant-expression :
statement(s);
break;
case constant-expression :
statement(s);
break;
/* you can have any number of case statements*/
default : /*Optional*/
statement(s);
Copyright- *All rights reserved

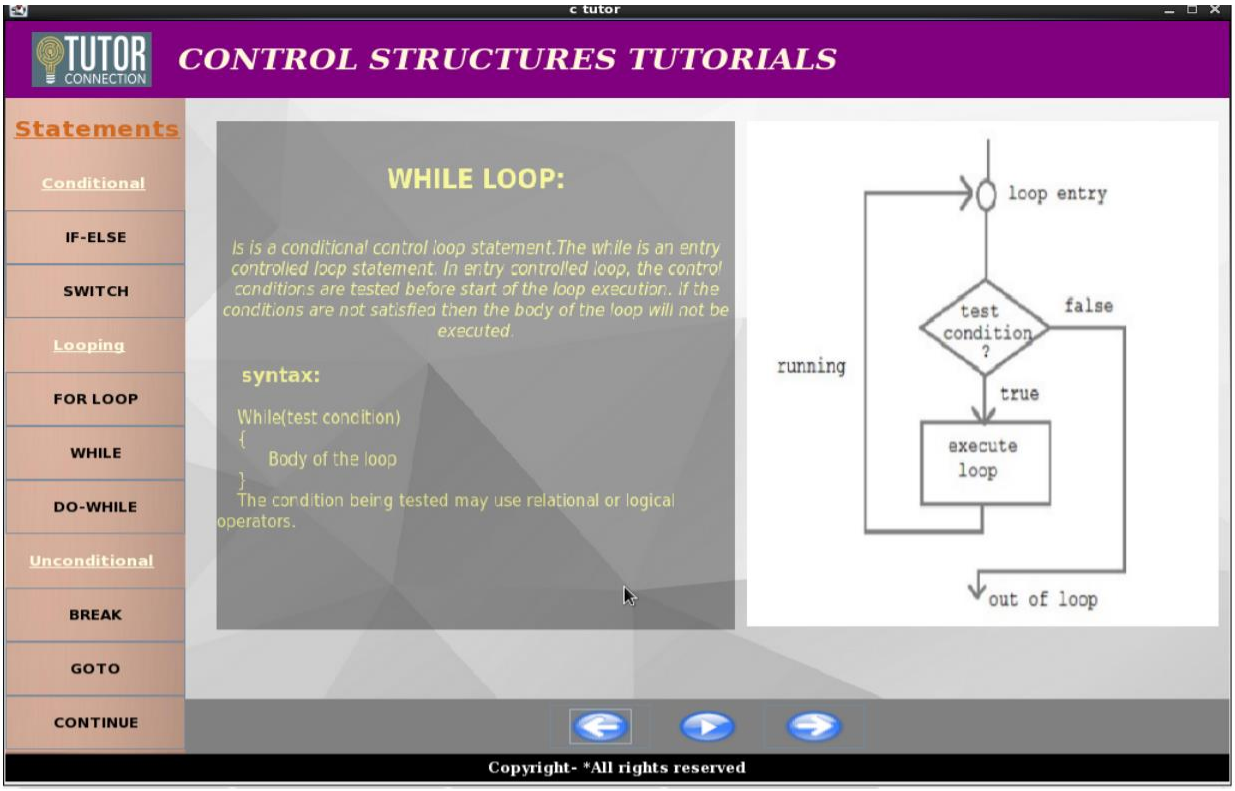

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
switch Animation
Sample Code:
#include<stdio.h>
void main()
{
int i=2;
Enter i
switch(i)
{
case 1: printf('In case 1');
break;
case 2:printf('In case 2');
break;
case 3: printf('In case 3');
break;
default: printf('In default');
}}
i
2
Conditions
case 2 ; True
Output
in case 2
case.
statement(s);
Copyright- *All rights reserved

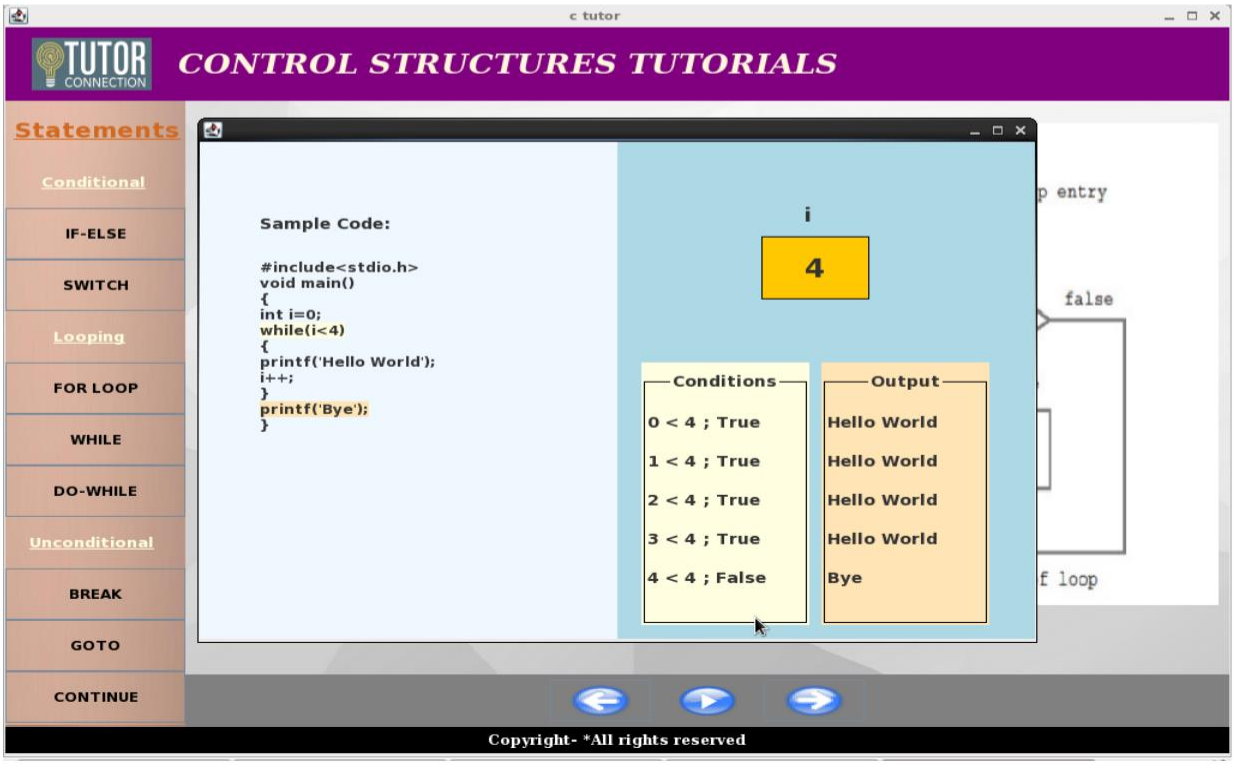

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
switch Animation
Sample Code:
#include<stdio.h>
void main()
{
int i=2;
Enter i
switch(i)
{
case 1: printf('In cas
break;
case 2:printf('In cas
break;
case 3: printf('In case 3');
break;
default: printf('In default');
}}
Input
Enter i
4
OK
Cancel
i
2
case.
Output
in case 2
statement(s);
Copyright- *All rights reserved

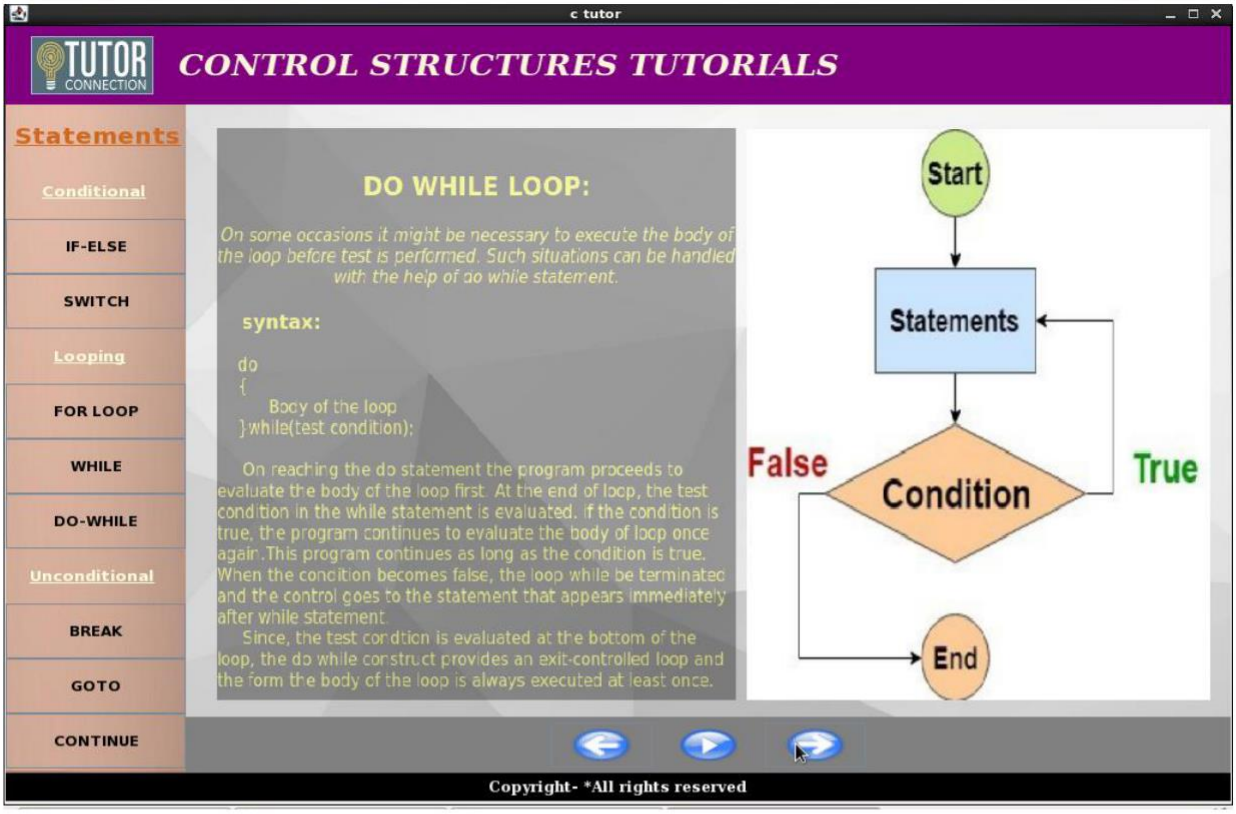

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
switch Animation
Sample Code:
#include<stdio.h>
void main()
{
int i=2;
Enter i
switch(i)
{
case 1: printf('In case 1');
break;
case 2:
break;
case 3:printf('In case 3');
break;
default: printf('In default');
}}
i
4
case.
Conditions
default; True
Output
In default
statement(s);
Copyright- *All rights reserved

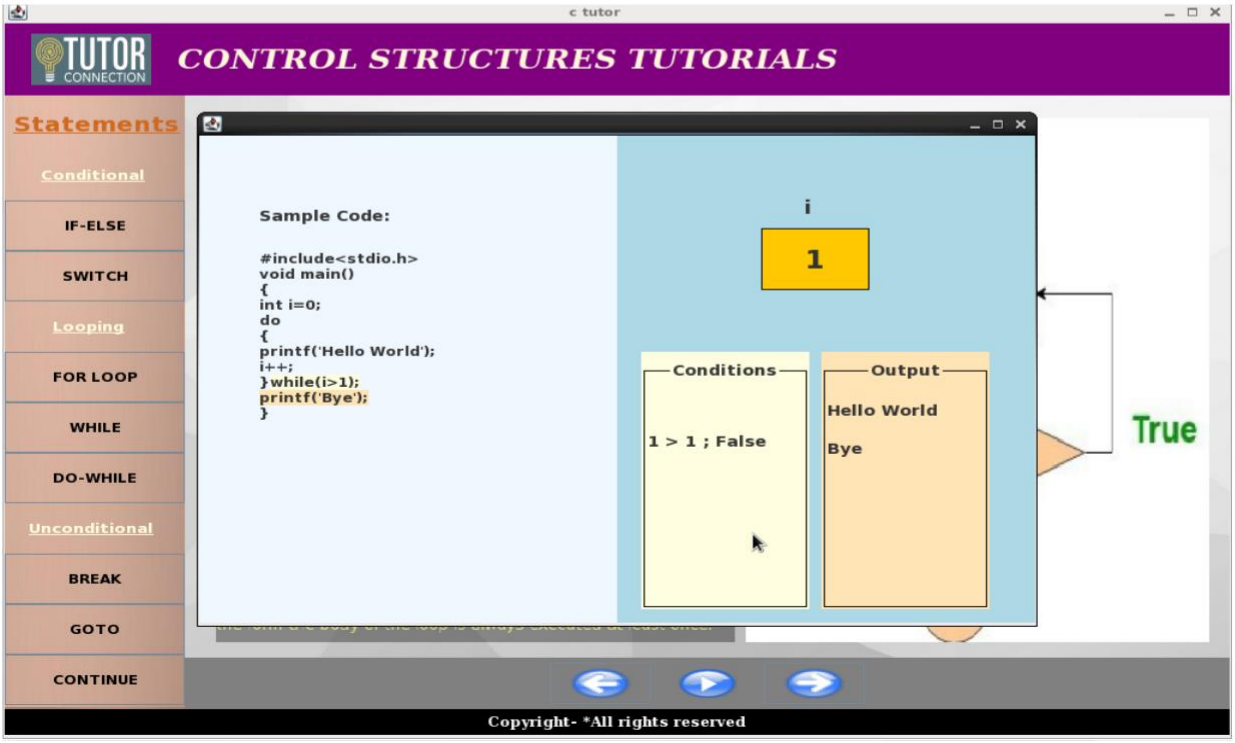

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
FOR LOOP:
The for loop is another entry controlled loop that provides a more concide loop control structure.
syntax:
for(initialization; test condition; increment)
{
Body of the loop
}
1(Initialization):
Here, the intialization of control variable takes place using assignment operator.
2(Test condition):
The values of control is tested using the test condition. If condition is true body of loop is executed else it goes of loop.
3(Increment):
here, the value of control variable is incremented (Ex. i++ or i=i+c;c=constant).
Start
Initialization
False
Condition
Increment / Decrement
True
Statements
End
Copyright- *All rights reserved

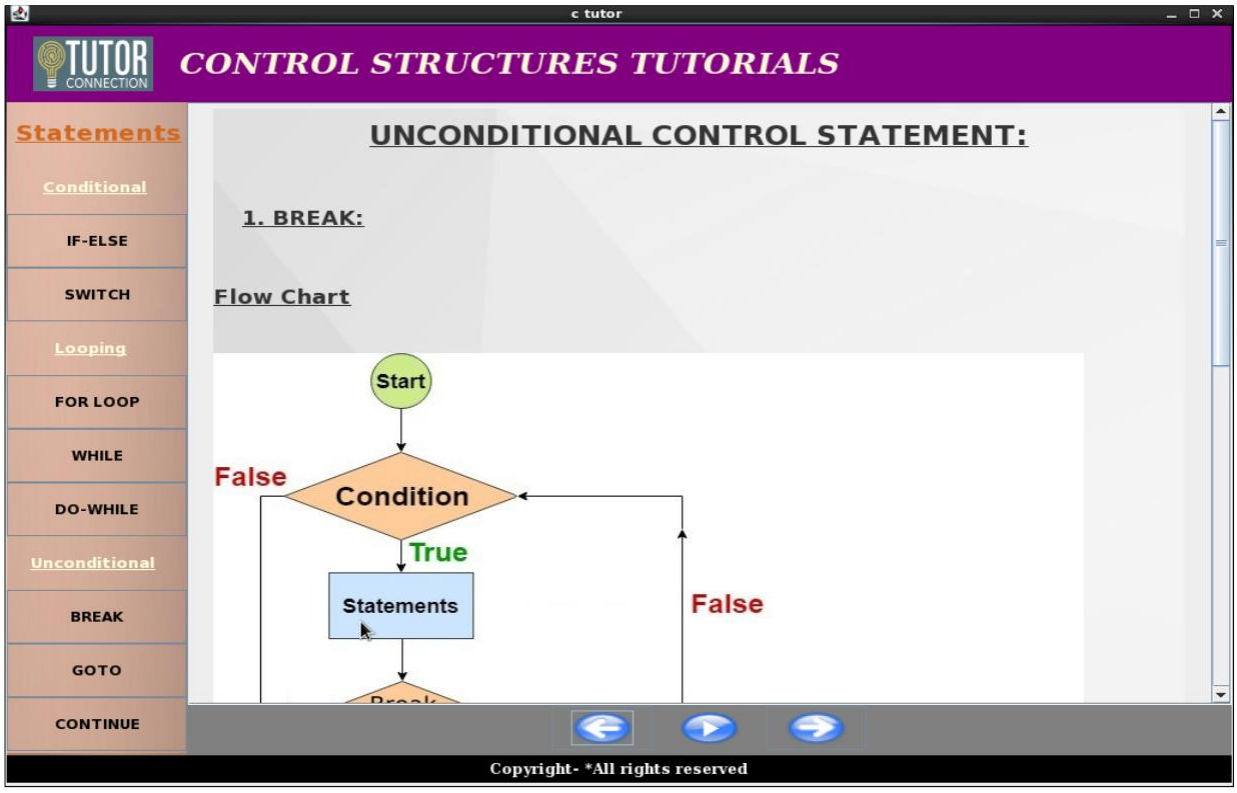

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
Sample Code:
#include<stdio.h>
void main()
{
int x=4;
for(int i=0;i<x; i++)
{
printf('Hello World');
}
printf('Bye');
}
i
4
Conditions
0 < 4 ; True
1 < 4 ; True
2 < 4 ; True
3 < 4 ; True
4 < 4 ; False
Output
Hello World
Hello World
Hello World
Hello World
Bye
Increment / Decrement
Copyright- *All rights reserved

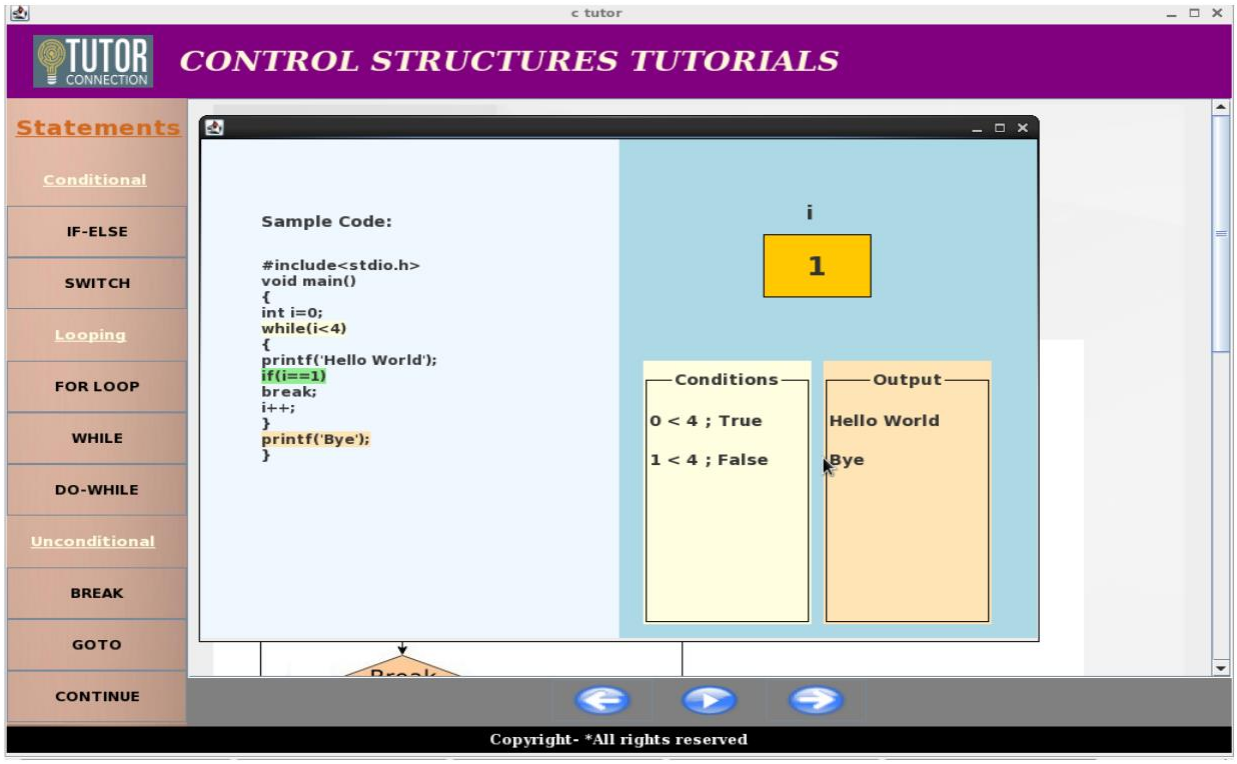

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
WHILE LOOP:
Is is a conditional control loop statement.The while is an entry controlled loop statement. In entry controlled loop, the control conditions are tested before start of the loop execution. If the conditions are not satisfied then the body of the loop will not be executed.
syntax:
While(test condition)
{
Body of the loop
}
The condition being tested may use relational or logical operators.
loop entry
test condition ?
false
true
execute loop
running
out of loop
Copyright- *All rights reserved

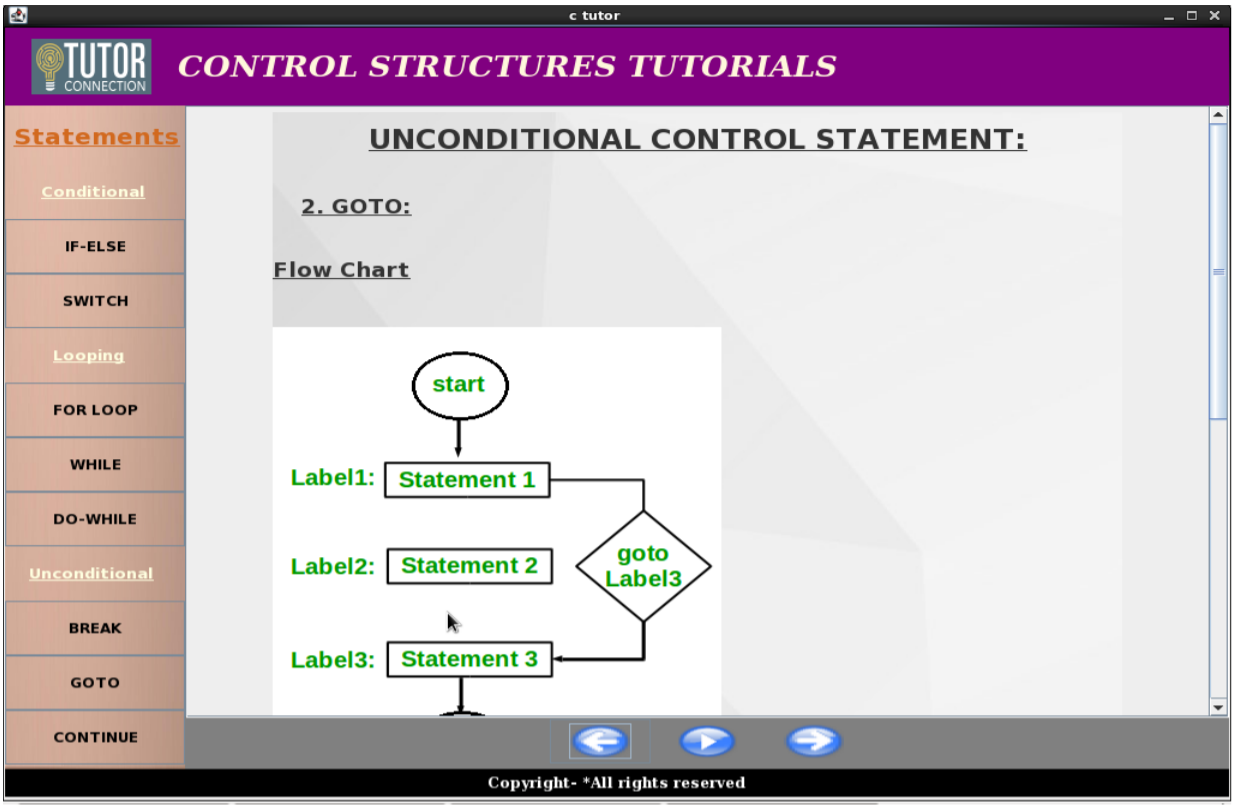

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
Sample Code:
#include<stdio.h>
void main()
{
int i=0;
while(i<4)
{
printf('Hello World');
i++;
}
printf('Bye');
}
i
4
Conditions
0 < 4 ; True
1 < 4 ; True
2 < 4 ; True
3 < 4 ; True
4 < 4 ; False
Output
Hello World
Hello World
Hello World
Hello World
Bye
Copyright- *All rights reserved

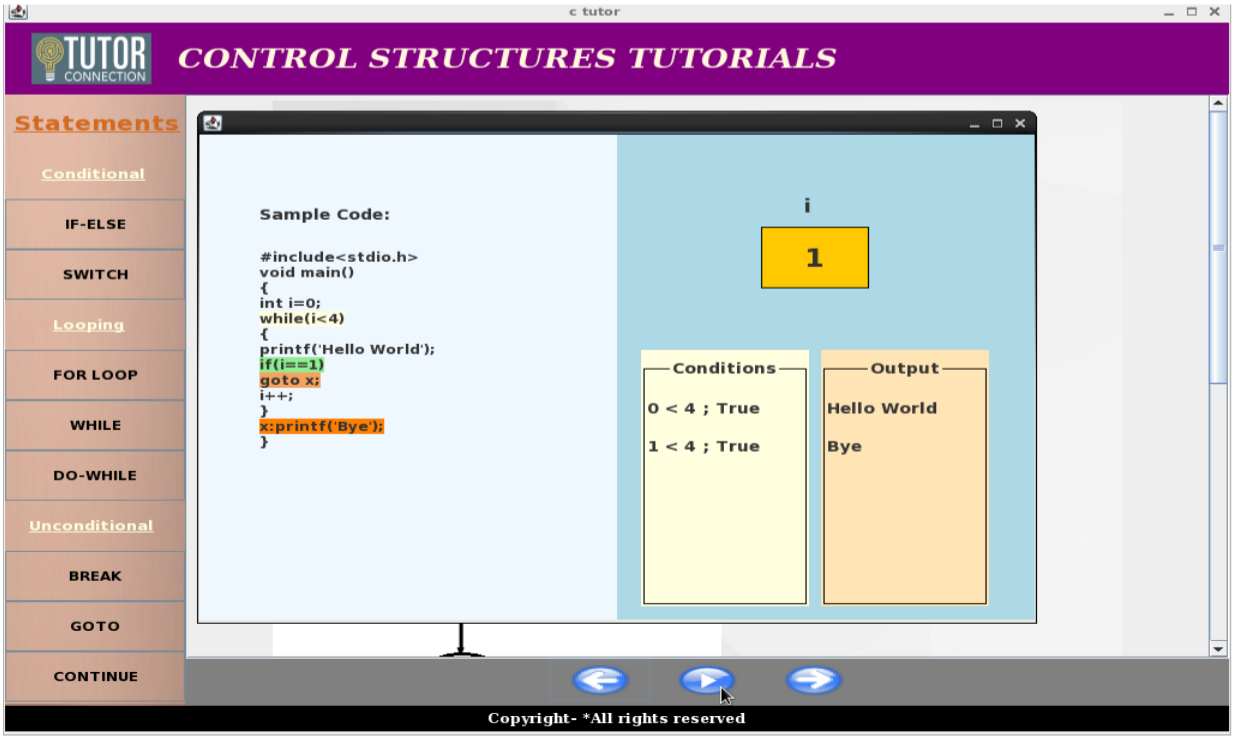

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
DO WHILE LOOP:
On some occasions it might be necessary to execute the body of the loop before test is performed. Such situations can be handled with the help of do while statement.
syntax:
do
{
Body of the loop
}while(test condition);
On reaching the do statement the program proceeds to evaluate the body of the loop first. At the end of loop, the test condition in the while statement is evaluated. If the condition is true, the program continues to evaluate the body of loop once again.This program continues as long as the condition is true. When the concition becomes false, the loop while be terminated and the control goes to the statement that appears immediately after while statement
Since, the test cordtion is evaluated at the bottom of the loop, the do while construct provides an exit-controlled loop and the form the body of the loop is always executed at least once.
Start
Statements
False
Condition
True
End
Copyright- *All rights reserved

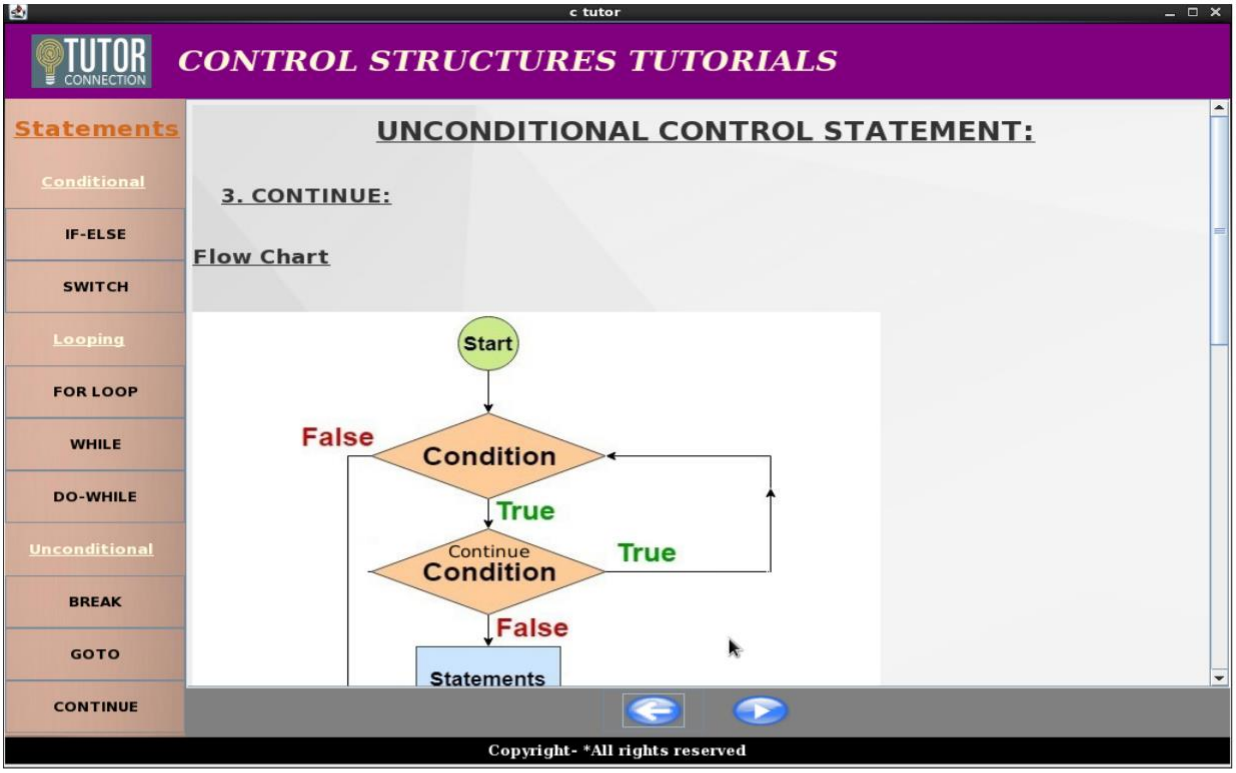

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
Sample Code:
#include<stdio.h>
void main()
{
int i=0;
do
{
printf('Hello World');
i++;
}while(i>1);
printf('Bye');
}
i
1
Conditions
1 > 1 ; False
Output
Hello World
Bye
True
Copyright- *All rights reserved

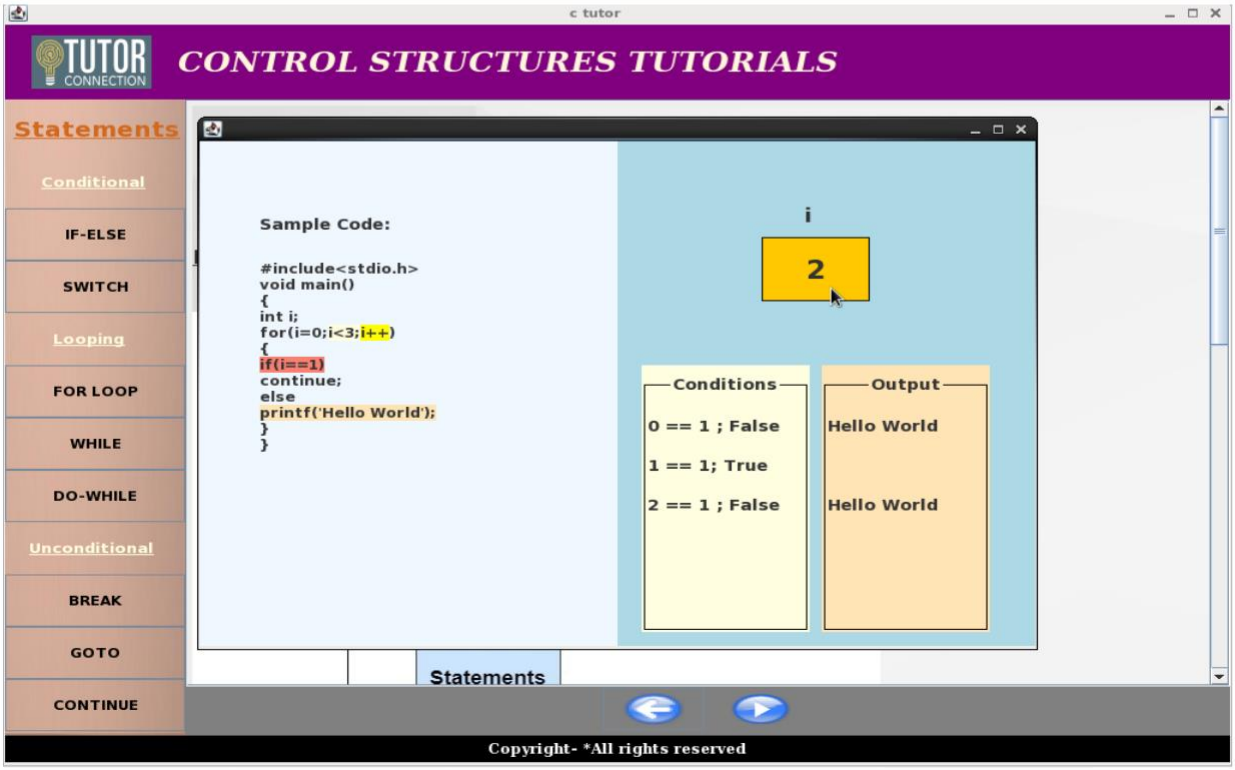

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
UNCONDITIONAL CONTROL STATEMENT:
1. BREAK:
Flow Chart
Start
False
Condition
True
Statements
False
Copyright- *All rights reserved


c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
Sample Code:
#include<stdio.h>
void main()
{
int i=0;
while(i<4)
{
printf('Hello World');
if(i==1)
break;
i++;
}
printf('Bye');
}
i
1
Conditions
0 < 4 ; True
1 < 4 ; False
Output
Hello World
Bye
Copyright- *All rights reserved


c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
UNCONDITIONAL CONTROL STATEMENT:
2. GOTO:
Flow Chart
start
Label1:
Statement 1
Label2:
Statement 2
goto Label3
Label3:
Statement 3
Copyright- *All rights reserved


c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
Sample Code:
#include<stdio.h>
void main()
{
int i=0;
while(i<4)
{
printf('Hello World');
if(i==1)
goto x;
i++;
}
x:printf('Bye');
}
i
1
Conditions
0 < 4 ; True
1 < 4 ; True
Output
Hello World
Bye
Copyright- *All rights reserved

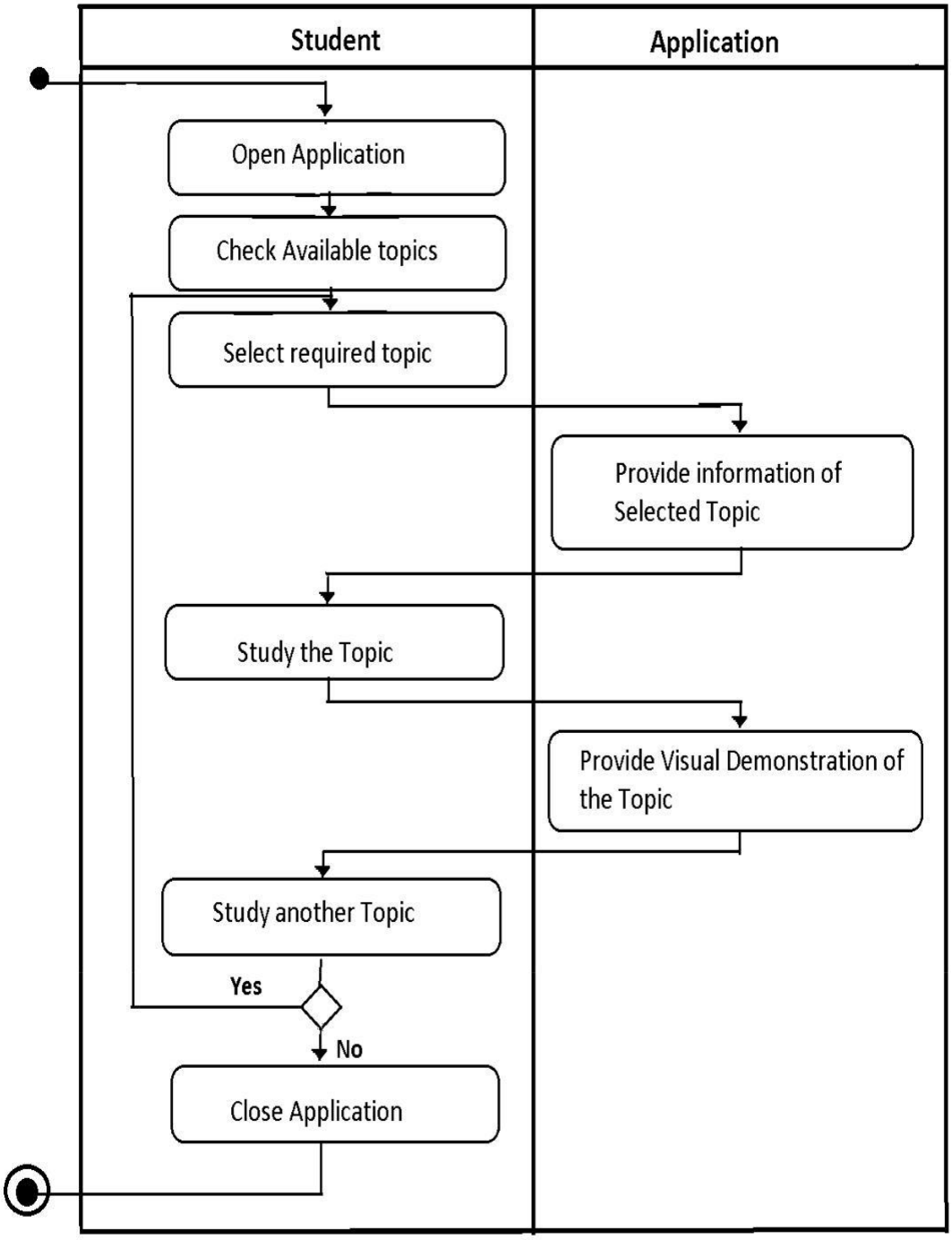

c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
UNCONDITIONAL CONTROL STATEMENT:
3. CONTINUE:
Flow Chart
Start
False
Condition
True
Continue
Condition
True
False
Statements
Copyright- *All rights reserved


c tutor
TUTOR CONNECTION
CONTROL STRUCTURES TUTORIALS
Statements
Conditional
IF-ELSE
SWITCH
Looping
FOR LOOP
WHILE
DO-WHILE
Unconditional
BREAK
GOTO
CONTINUE
Sample Code:
#include<stdio.h>
void main()
{
int i;
for(i=0;i<3;i++)
{
if(i==1)
continue;
else
printf('Hello World');
}
}
i
2
Conditions
0 == 1 ; False
1 == 1; True
2 == 1 ; False
Output
Hello World
Hello World
Statements
Copyright- *All rights reserved

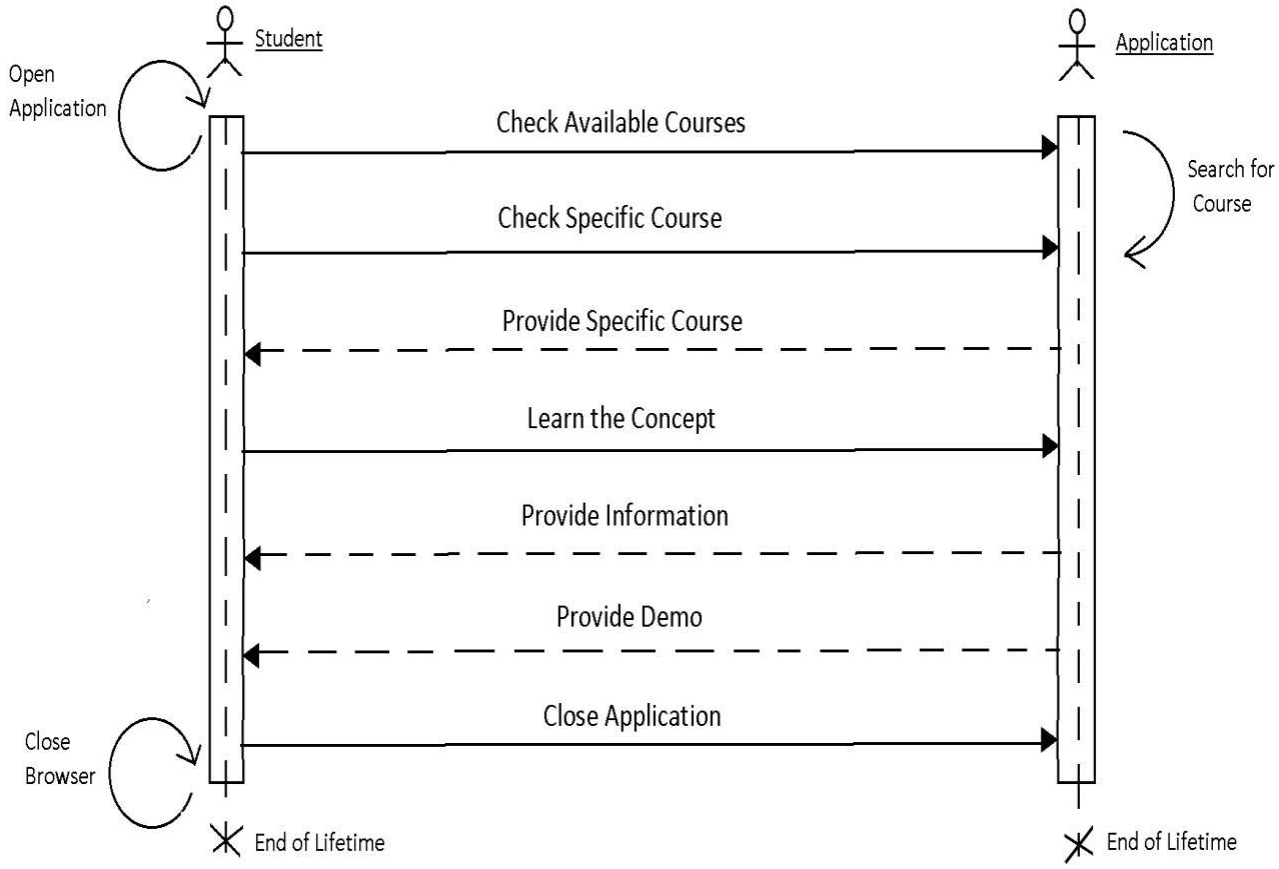
4. UML Diagram

4.1 Use Case Diagram

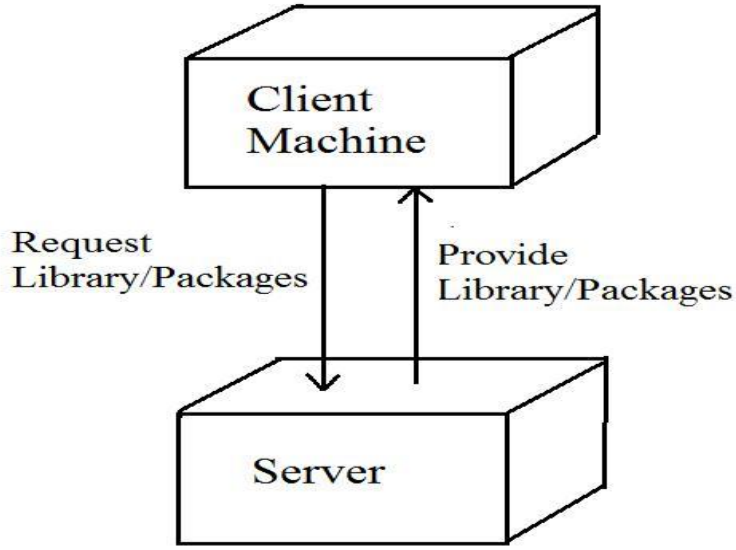
4.2 Activity Diagram



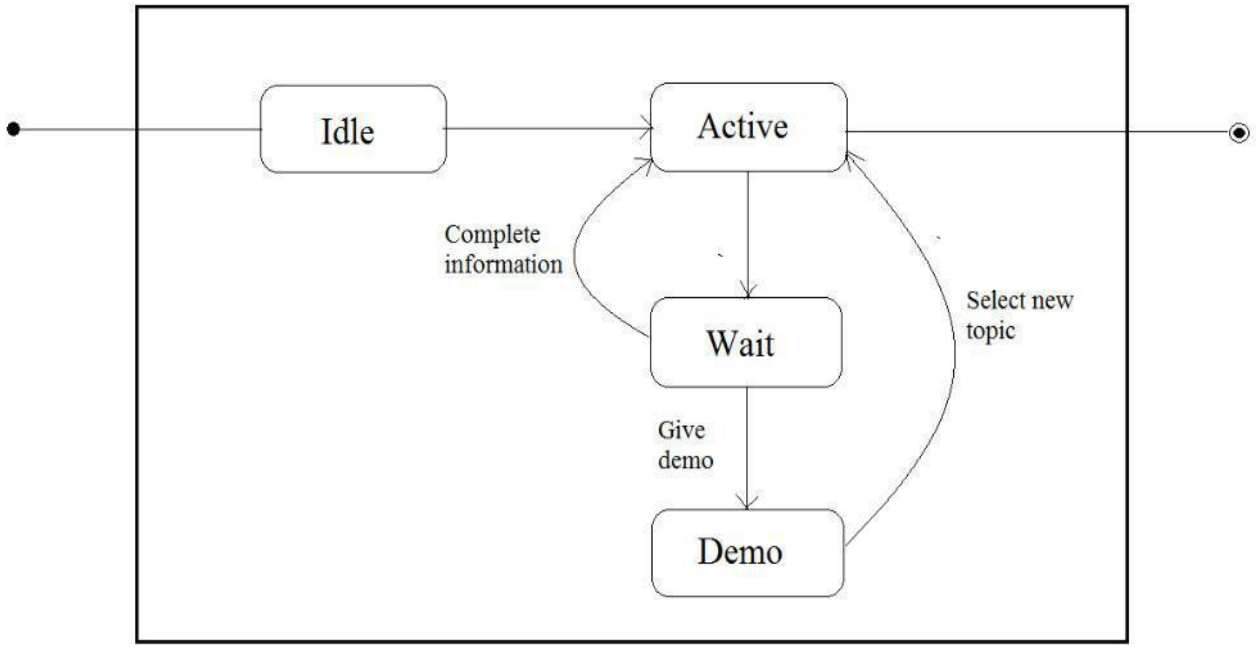
4.3 Sequence Diagram



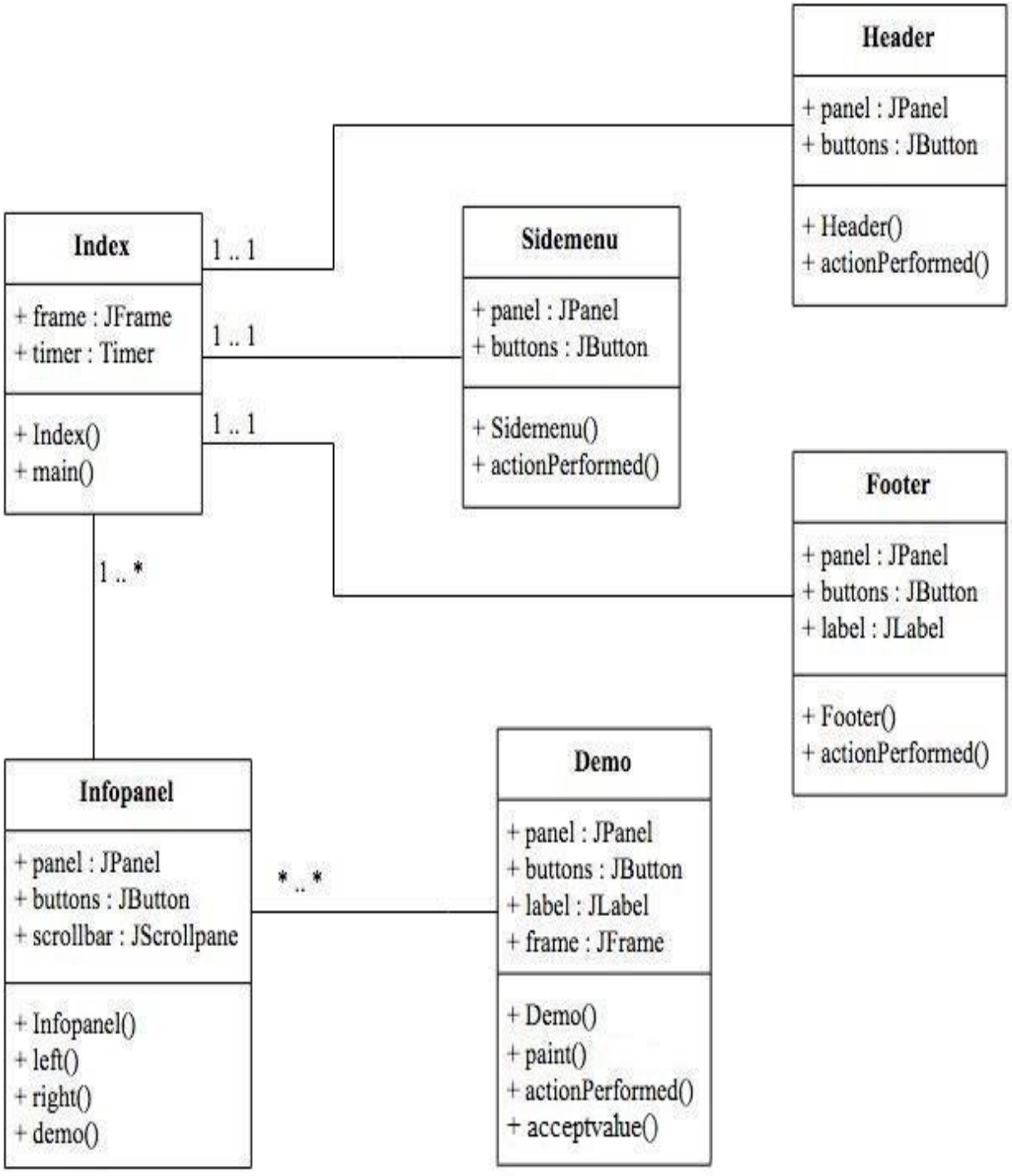
4.4 Deployment Diagram



4.5 State Diagram



4.6 Class Diagram



5. CODING

5.1 Hardware Specification

We were provided with all the software as well as hardware facilities by the college at our lab itself.

Following are some of the hardware specifications provided to us at our lab.

- **5.1.1. Client side configurations:**

  Operating System: FEDORA 33

  Release: 33

  Kernel: 5.11.21-200.fc33.x86_64

  GNOME version:3.38.5

  OS type: 64 bit

  Memory: 3.7 GiB

  Processor: Intel core i3 5^{th} generation

  RAM: 256 MB

  Internal hardisk (HDD): 40 GB

  CPU: 2.00GHz

- **5.1.1. Server side configurations:**

  Operating System: Linux Redhat 6.0 Enterprise

  OS type: 64 bit

  RAM: 256 MB

  Processor: Intel Xeon E7-8855 V4

  Hardisk (HDD): 20 GB

- **Switch box:**

  A network switch (also called switching hub, bridging hub, officially MAC bridge[1]) is a computer networking device that connects devices together on a computer network by using packet switching to receive, process, and forward data to the destination device.

  The switch connects all the computers together to the server using **TREE** topology.

5.2 Platform

CentOS 6.0 (in built n software's are C, C++, Java, PHP, Databases as PSQL, MySQL etc packages).

5.3 Programming Languages Used

Given below are some of the web technologies used for developing the website.

- **JAVA :**

Java is one of the most popular and widely used programming language and platform. A platform is an environment that helps to develop and run programs written in any programming language. Java is fast, reliable and secure. From desktop to web applications, scientific supercomputers to gaming consoles, cell phones to the Internet, Java is used in every nook and corner.

The fundamentals of Java came from a programming language called C++. Although C++ is a powerful language, it is complex in its syntax and inadequate for some of Java's requirements. Java built on and improved the ideas of C++ to provide a programming language that was powerful and simple to use.

➢ **Swing**

Java Swing is a part of Java Foundation Classes (JFC) that is used to create window-based applications. It is built on the top of AWT (Abstract Windowing Toolkit) API and entirely written in java. Unlike AWT, Java Swing provides platform-independent and lightweight components. The javax.swing package provides classes for java swing API such as JButton, JTextField, JTextArea, JRadioButton, JCheckbox, JMenu, JColorChooser etc.

5.4 Coding Style followed

- **Layout/Structure Design :**

  The technology used here for describing the basic structure/layout of the application is Swing.

  Swing is a set of program components for Java programmers that provide the ability to create graphical user interface (GUI) components, such as buttons and scroll bars, that are independent of the windowing system for specific operating system . Swing components are used with the Java Foundation Classes (JFC).

- **For Demonstration :**

  The technology used here for showing the basic demonstration of the application is Multithreading.

  Multithreading in java is a process of executing multiple threads simultaneously. A thread is a lightweight sub-process, the smallest unit of processing. Multiprocessing and multithreading, both are used to achieve multitasking. However, we use multithreading than multiprocessing because threads use a shared memory area. They don't allocate separate memory area so saves memory, and context-switching between the threads takes less time than process. Java Multithreading is mostly used in games, animation, etc.

6. Testing

6.1. Test Cases and Test Results

Test Case	Test Result	Expected Result
Input value (Value of x)	Value entered by user should be numeric else display input dialog box again.	Value entered by user should be numeric and demo should be performed based on that value.
Frame bounds	Open demonstration frame.	Control structure demo should not exceed the frame size.
Threads	Repeated same demo.	Smooth demo starting from beginning without interfering other threads.
Selection of Topic	Browser side menu click on required topic.	Information of the selected topic displayed on the panel.

7. Limitations and Future Enhancements

Though the application is user-friendly to its users, some intricate options could not be covered into it; partly because of logistic and partly due to lack of sophistication. Paucity of time was also major constraint, thus it was not possible to make the software foolproof and dynamic.

Considerable efforts have made the software easy to operate even for the people not related to the field of computers but it is acknowledged that a layman may find it a bit problematic at the first instance.

It can be summarized that the future scope of the project circles around maintaining information regarding:

- We can add printer in future.
- We will host the platform on online servers to make it accessible worldwide.
- We can add different features in this system to make it more user-friendly.
- We can conduct quiz based on topics.

The above mentioned points are the enhancements which can be done to increase the applicability and usage of this project.

8. Conclusion

It is the user-friendly application for Students who are not able to attend lectures for some reasons and for those who can't go to colleges/school. Several user-friendly coding have also adopted. This project will be useful for many students; they can rewatch any topic in between if they wish to. This package shall prove to be a powerful package in satisfying all the requirements of the school/college.

At the end it is concluded that we have made effort on following points…

- A description of the background and context of the project and its relation to work already done in the area.
- Made statement of the aims and objective of the project.
- We define the problem on which we are working in the project.
- We describe the requirement Specification of the system and the actions that can be done on these things.
- We included features and operations in detail, including demonstration.
- Finally the system is implemented and tested according to test cases.