SIX WEEKS SUMMER TRAINING

REPORT

On

GRAPHICAL IMPLEMENTATION OF DATA STRUCTURES AND ALGORITHMS

Submitted by

S S Mercy Queen

11901859

BTech (Computer Science and Engineering)

Under the Guidance of

Mr. Sandeep Jain

(CEO of GeeksForGeeks)

School of Computer Science & Engineering

Lovely Professional University, Phagwara

(June – July, 2021

DECLARATION

I hereby declare that I have completed my six weeks summer training at GeeksForGeeks from May,22,2021 to July, 27, 2021 under the guidance of Mr. Sandeep Jain I have declare that I have worked with full dedication during these six weeks of training and my learning outcomes fulfill the requirements of training for the award of degree of BTech (Computer Science and Engineering), Lovely Professional University, Phagwara.

Date : 27 July 2021 S S Mercy Queen

11901859

ACKNOWLEDGEMENT

First I would like to thank the Department of Computer Science and Engineering, Lovely Professional University, Phagwara for giving me the opportunity to do an internship in the form of MOOC, so that we could complete our internship even in this pandemic and also for providing sufficient amount of time for completion of the summer internship.

I would also like to thank GeeksForGeeks for providing us with such a MOOC to get in depth and basic knowledge required in our career.

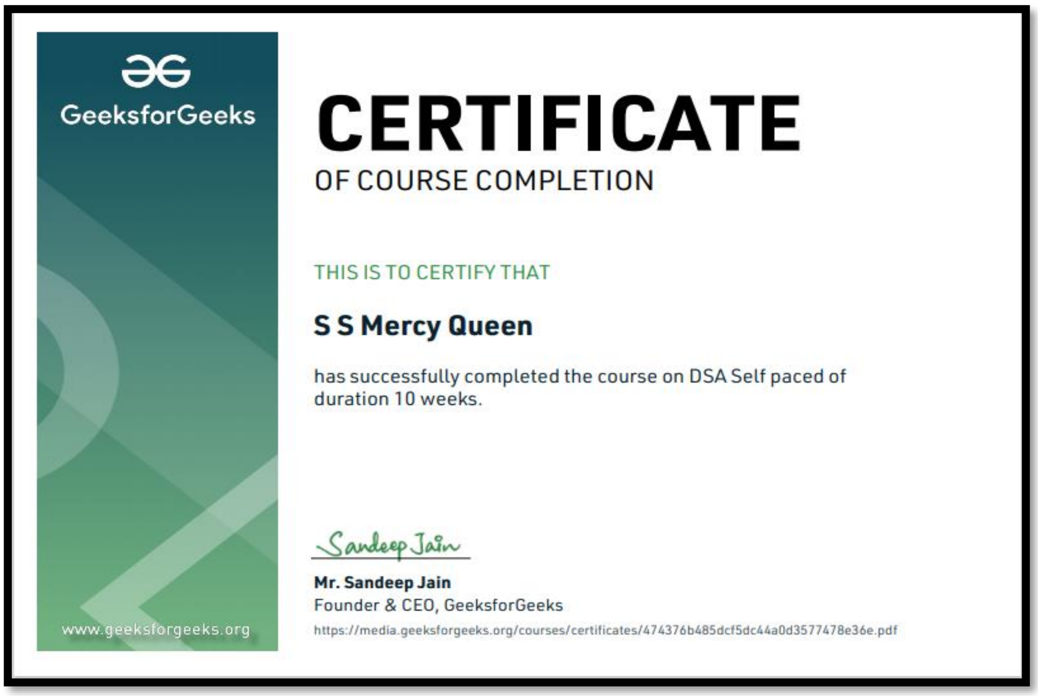
I would like to thank Mr. Sandeep Jain, CEO of GeeksForGeeks and the instructor of the MOOC, for making us understand the course in an efficient manner. He helped us to understand the concepts clearly by using as much examples as possible. The quizzes, coding questions and contests have helped me a lot in improving myself.

Lastly, I am extremely great full to my department staff members who helped me in successful completion of this internship and the mini project.

S S Mercy Queen

11901859

CERTIFICATE



GeeksforGeeks

CERTIFICATE

OF COURSE COMPLETION

THIS IS TO CERTIFY THAT

S S Mercy Queen

has successfully completed the course on DSA Self paced of duration 10 weeks.

Sandeep Jain

Mr. Sandeep Jain

Founder & CEO, GeeksforGeeks

www.geeksforgeeks.org

https://media.geeksforgeeks.org/courses/certificates/474376b485dcf5dc44a0d3577478e36e.pdf

TABLE OF CONTENT

LIST OF FIGURES

CHAPTER 1

INTRODUCTION OF THE WORK

- ABOUT THE COMPANY :

There is a famous saying by Marissa Mayer. “Geeks are people who love something so much that all the details matter”. Many have tried to define who these geeks are, but this one is right in spot! For a geek, all that is required to keep him busy and happy, is a group of other geeks working on the same problem trying to figure out some answer. How many times were you frustrated while looking out for a good collection of programming/algorithm/interview questions? What did you expect and what did you get? This is where GeeksforGeeks comes into picture – A computer science portal for geeks, by geeks.

GeeksforGeeks is a company situated in Noida. It was found and created by Mr. Sandeep Jain with a goal in mind to provide well written, well explained and well thought solutions for selected questions . The core team of five supper geeks constituting of technology lovers and computer science enthusiasts have been constantly working in this direction.

GeeksforGeeks believes in the power of experience. That is why it interviews pro – coders and shares those words of wisdom with everyone.

- ABOUT THE COURSE:

DSA – self paced course is divided into 8 weeks where I learnt the basics of DSA and practiced questions and attempted the assessment tests. This could further help me in preparing for interviews with top – notch companies. This course does not require any prior knowledge of Data Structure and Algorithms, but a basic knowledge of any programing language.

INTRODUCTION OF PROJECT UNDERTAKEN

- ABOUT THE MINI – PROJECT:

  Basic Data Structures like Arrays, Linked list and basic Algorithms like insertion and deletion are easy to understand but the intermediate and advanced Data Structures and Algorithms like stacks, queues, trees and sorting algorithms are difficult to understand.

  So to make it easier practical or a visual approach could be used. That is what is done in the mini – project – Graphical implementation of data structures and algorithms.

  This project called – 'Graphical implementation of Algorithms and data structures' is built in order to solve the most common problem a student faces when he is enrolled into the field of computer science and engineering. In fact, I myself have faced this issue of not able to visualizing the working of the data structures and algorithms. We have been taught about how to code for it but we as students, face difficulty in understanding. So I came up with this mini-project. Although it does not contain all the algorithms and data structures but I have made an attempt to work with some of the basic and known ones.

  This project is basically a desktop application completely coded in JAVA and for the Graphics Part Java Swing is used. When the user open this application then initially there is a splashing screen then he gets into the home page where he can find two option, either he can go to the about page where there is a description about the project, or he can go to the get started page from where he can choose either data structure section or algorithm section.

  Inside the data structure section, three data structures namely, Stack, Queue and Trees (traversals) are described.

Inside the algorithms section, four algorithms namely, Selection sort, Insertion Sort, Bubble sort and Binary Search are described.

And in each part of these sections there is an “i” icon, clicking on which will open up a pop up box which has the code for the data structure or algorithms implemented respectively.

- DATA STRUCTURE :

The name itself indicates its meaning. It means organizing the data in a very structured manner. There are many ways of organizing the data in the memory, for example – Array. It is a collection of memory elements in which data is stored sequentially, that is one after another. In other words, we can say that array stores the elements in a continuous/contiguous manner. This organization of data is done with the help of an array of data structures. There are also other ways to organize the data in memory.
Data Structure is broadly classified into two types : primitive and non – primitive data structure. Primitive data structure are the ones which are in built like integer, float and many other. Non -primitive is the combination of many primitive data structure, example array, linked list and so on.

- ALGORITHM:

An algorithm is a process, or a set of rules required to perform calculations or some other problem-solving operations especially by a computer. The formal definition of an algorithm is that it contains the finite set of instructions which are being carried in a specific order to perform the specific task. It is not the complete program or code; it is just a solution (logic) of a problem, which can be represented either as an informal description using a Flowchart or Pseudocode.

Algorithms can also be classified , like Data Structures, into brute force, divide and conquer, greedy and dynamic programming.

REASON FOR CHOOSING THE TECHNOLOGY

Although there are a lot many new technologies like Machine Learning, Data Science and many more to learn but I chose Data Structure and Algorithm because it is the foundation of many new technologies. If you get a grip on this foundation then programming becomes very much easier. There are a lot of applications of this technology, some of which are mentioned below:

- APPLICATION OF ARRAYS:

  Arrangement of leader – board of a game can be done simply through arrays to store and arrange them in descending order to clearly make out the rank of each player in the game.
  A simple question paper is an array of numbered questions with each of them assigned to some marks.

- APPLICATION OF LINKED LIST:

  Images are linked with each other using linked list in an image viewer.
  Music players also use the same technique to switch between music.
  Next and previous URL in a webpage can also be accessed using linked list.

- APPLICATION OF GRAPH:

  A graph is a data structure that consists of nodes and vertices. This concept is used in many social media platforms. In social media networking sites like Facebook, Instagram and many other, each user is a node, and when a user is a follower of another user then there is an edge between them. This is how graph concept is used here.

- APPLICATION OF DYNAMIC PPROGRAMMING:

  The concept of Dynamic Programming is used in various fields. It is used in speech recognition, image processing, financial trading ana many more.

- APPLICATION OF BACKTRACKING:

  For solving famous puzzles like N-queens, crosswords, Sudoku we use Backtracking.
  It is also used in solving various optimization and constraint satisfaction problems.

- APPLICATION OF NUMBER THEORY:

  Used in cryptography and computer graphics.
  For designing Hash functions and random number generators.

- APPLICATION OF TRAVELLING SALESMAN PROBLEM:

  The famous problem travelling salesman is used in vehicle routing and is used by many applications like Google Maps.

PROFILE OF THE PROBLEM

Data Structure and Algorithm is a concept which is very difficult to understand. Below are some of the points listed which makes it clear why understanding them is difficult.

- INTRODUCTION TO NEW TCHNOLOGIES:

  There are a lot of programming language around us like, C++, C, C#, Java, Python, and a lot of editors as well so students get confused which language and editor to use. And they spend a lot of time thinking about this. So it leads to procrastination. To have proper learning of Data Structure and algorithms, programmers have to be updated about various new technologies that are coming up almost every day.

- UNABLE TO GET THE DOUBTS CLEARED:

  While we know that it is challenging to learn Data Structures, programmers need certain support and guidance whenever they get stuck in problem – solving. Although the support system at most of the universities is good but the student hesitates to ask the doubts hence their doubts remain as doubts forever.

- LACK OF PATIENCE:

  While learning data structures and algorithms, programmers tend to forget that data is gathered and then fed to train the algorithms. The results of how well an algorithm is performing might take time. Most of the programmers find this entire process daunting and develop a sense of negativity. However, it’s important to know, for anything, to show results, we need to be patient and trust the process.

- OVERLAPPING TOPICS:

  Subjects in data structure and algorithms are interconnected to each other. Most freshers in programming tend to get muddled with subjects and sometimes learn the advanced ones before the basic ones. Here, it is always better to build a house when the foundation is strong. So, follow the standard curriculum that enlists topics in proper order.

- UNCLEAR METHOD OF TEACHING:

  This is one of the most common reason why students don’t understand data structure and algorithm. Educators use hose methods of teaching which seems to be a bit complex to grab. Most of them skip topics and move ahead. Educators must find out an easy way to help programmers to understand better by giving real life implementation of them.

EXISTING SYSTEM AND TECHNOLOGY

- EXISTING SYSTEM:

  In order to make us data structures understand better universities make such an Instruction Plan in which after learning the concept implementation of the same is done practically like prompting input from the user and giving the output for the same.

- IMPROVEMENTS THAT CAN BE MADE TO IT:

  In order to understand it better, educators can make us build small real - life projects, so that we can get an idea how to implement it.

PROBLEM ANALYSIS

- PRODUCT DEFINITION:

The mini – project is completely built using Java and Java Swing and netbeans 8.2 editor. This project has two part like the course, one part , to understand the algorithms and the other one to understand the data structure. Under the Data Structure part below listed data structures are explained and implemented:

- Stack
- Queue
- Tree

In the other part , that is the Algorithms part below listed algos are explained.

- Bubble Sort
- Selection Sort
- Insertion Sort
- Binary Search

With each Jframe there is an icon clicking on which can display another jframe which has the pseudo code behind the implementation.

In this project the user will enter the elements he wants to sort , suppose, then the user can not only see the final output but also he can see the process by which the array gets sorted.

- FEASIBILITY ANALYSIS:

  The project is about graphically representing the implementation of Data Structures and Algorithms. This can be done either using Python (tkinter) , C++ (OpenGL) or by using animations or using Java and Java Swing. So I chose to do it with Java Swing.

  Reason behind choosing Java Swing : It is user friendly and programming oriented.

  Hence choosing Java for the projects seems to be more feasible.

SOFTWARE REQUIREMENT ANALYSIS

- SYSTEM CONFIGURATIONS:

The software requirement specification can produce at the culmination of the analysis task. The function and performance allocated to software as part of system engineering are refined by established a complete information description, a detailed functional description, a representation of system behavior, and indication of performance and design constrain, appropriate validate criteria, and other information pertinent to requirements.

- SOFTWARE REQUIREMENTS:

• Operating System	: Windows 8 and onwards
• Coding Language	: Java
• Front – End	: Java Swing

CHALLENGS FACED WHILE CREATING THIS PROJECT

As I have not worked much with Java Swing so I had problem in the UI part. But I treed my best to understand and implement those things.

CHAPTER – 2

DESIGN

- FLOWCHART :

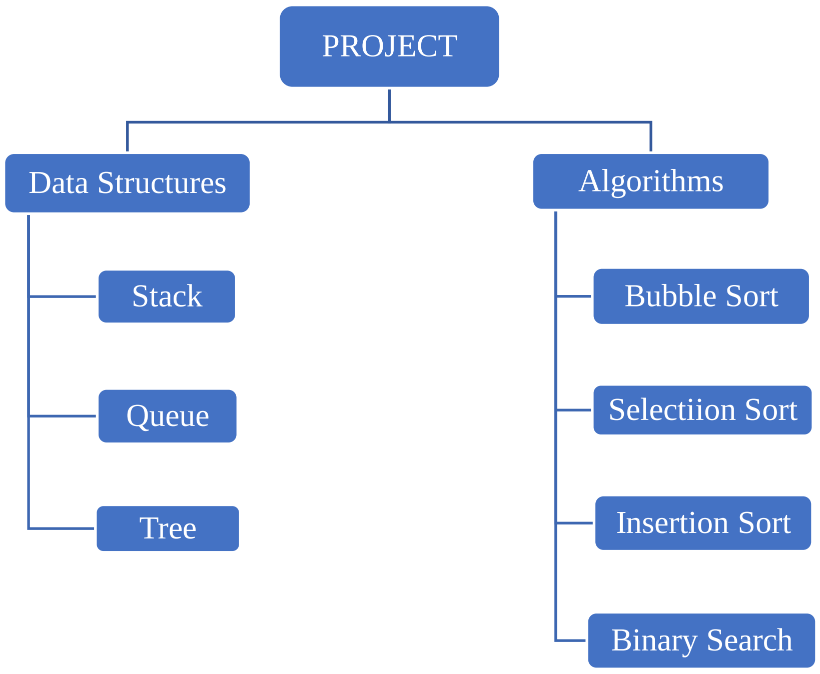


Fig. 2.1

- CODING :

  - LOADING SCREEN:

```
public static void m(loadingscreen ls)
{try
{for(int i=0;i<=100;i++)
{Thread.sleep(100);
ls.LoadingValue.setText(i+ " %");
if(i==10  ls.LoadingLabel.setText("Turning On Modules...");
if(i==20) ls.LoadingLabel.setText("Loading Modules...");
if(i==50 ls.LoadingLabel.setText("Connecting to Server...");
if(i==70)ls.LoadingLabel.setText("Connection Successful!");
if(i==80) ls.LoadingLabel.setText("Launching Application...");
ls.jProgressBar1.setValue(i); }}
catch(Exception e) {
JOptionPane.showMessageDialog(null,e);}}
```

  - HOME:

```
 private void jLabel1MouseClicked(java.awt.event.MouseEvent
evt)
{exit(0);}
private void
jTextField1MouseExited(java.awt.event.MouseEvent evt)
{jTextField1.setForeground(new Color(204,255,255));}
private void
jTextField1MouseMoved(java.awt.event.MouseEvent evt)
{jTextField1.setForeground(new Color(111,185,143));}
private void jButton1MouseMoved(java.awt.event.MouseEvent
evt)
 { jButton1.setForeground(Color.white); }
```

```
private void jButton1MouseExited(java.awt.event.MouseEvent
evt) {
jButton1.setForeground(new Color(255,153,0));
private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt)
{one o=new one();o.setVisible(true); }
private void
jTextField1MouseClicked(java.awt.event.MouseEvent evt)
{About a=new About();a.setVisible(true);}
```

- ABOUT:

```
private void jLabel5MouseClicked(java.awt.event.MouseEvent
evt)
{this.dispose();}
```

- ALGO :

```
private void
jTextField1MouseMoved(java.awt.event.MouseEvent evt)
{jTextField1.setForeground(new Color(154,209,212));}
private void
jTextField1MouseExited(java.awt.event.MouseEvent evt)
{jTextField1.setForeground(Color.white); }
private void
jTextField3MouseMoved(java.awt.event.MouseEvent evt)
{jTextField3.setForeground(new Color(154,209,212)); }
private void
jTextField3MouseExited(java.awt.event.MouseEvent evt)
{jTextField3.setForeground(Color.white)’}
```

```
private void
jTextField1MouseClicked(java.awt.event.MouseEvent evt)
{one a=new one();a.setVisible(true);this.dispose();}
private void
jTextField3MouseClicked(java.awt.event.MouseEvent evt)
{DS a=new DS();a.setVisible(true);this.dispose();}
private void jButton1MouseMoved(java.awt.event.MouseEvent
evt) {
jButton1.setForeground(Color.white);   }
private void jButton1MouseExited(java.awt.event.MouseEvent
evt) {
jButton1.setForeground(new Color(0,50,73));  }
private void jButton2MouseExited(java.awt.event.MouseEvent
evt) {
jButton2.setForeground(new Color(0,50,73));   }
private void jButton2MouseMoved(java.awt.event.MouseEvent
evt) {
jButton2.setForeground(Color.white);  }
private void jButton3MouseExited(java.awt.event.MouseEvent
evt) {
jButton3.setForeground(new Color(0,50,73));  }
private void jButton3MouseMoved(java.awt.event.MouseEvent
evt) {
jButton3.setForeground(Color.white);
private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt)
{BubbleSort b=new BubbleSort();b.setVisible(true);
private void
jButton2ActionPerformed(java.awt.event.ActionEvent evt)
{SelectionSort c=new SelectionSort();c.setVisible(true);}
private void
jButton3ActionPerformed(java.awt.event.ActionEvent evt)
{LinearSearch y=new LinearSearch();y.setVisible(true);}
```

```
private void jLabel4MouseClicked(java.awt.event.MouseEvent
evt) {
NewJFrame o=new
NewJFrame();o.setVisible(true);this.dispose();
private void jButton4MouseMoved(java.awt.event.MouseEvent
evt) {
jButton4.setForeground(Color.white);}
private void jButton4MouseExited(java.awt.event.MouseEvent
evt) {
jButton4.setForeground(new Color(0,50,73));
private void
jButton4ActionPerformed(java.awt.event.ActionEvent evt)
{InsertionSort i=new InsertionSort();
i.setVisible(true);}
```

- DATA STRUCTURE:

```
private void jLabel1MouseClicked(java.awt.event.MouseEvent
evt) {
NewJFrame a=new NewJFrame();
a.setVisible(true);
this.dispose();}
private void
jTextField2MouseClicked(java.awt.event.MouseEvent evt)
{one o2=new one();
o2.setVisible(true); }
private void
jTextField1MouseClicked(java.awt.event.MouseEvent evt)
{Algo a=new Algo();
a.setVisible(true);
this.dispose();}
private void
jTextField2MouseExited(java.awt.event.MouseEvent evt)
{jTextField2.setForeground(Color.white);  }
private void
jTextField2MouseMoved(java.awt.event.MouseEvent evt)
{jTextField2.setForeground(new Color(154,209,212));  }
private void
jTextField1MouseMoved(java.awt.event.MouseEvent evt)
{jTextField1.setForeground(new Color(154,209,212));   }
private void
jTextField1MouseExited(java.awt.event.MouseEvent evt)
{jTextField1.setForeground(Color.white); }
private void
jButton3ActionPerformed(java.awt.event.ActionEvent evt)
{stackop a=new stackop();
a.setVisible(true);}
```

```
private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt)
{queueop a=new queueop();
a.setVisible(true);}
private void
jButton2ActionPerformed(java.awt.event.ActionEvent evt)
{treeop a=new treeop();
a.setVisible(true);}
private void jButton3MouseExited(java.awt.event.MouseEvent
evt) {
jButton3.setForeground(new Color(0,50,73));   }
private void jButton3MouseMoved(java.awt.event.MouseEvent
evt) {
jButton3.setForeground(Color.white);}
private void jButton1MouseExited(java.awt.event.MouseEvent
evt) {
jButton1.setForeground(new Color(0,50,73));      }
private void jButton1MouseMoved(java.awt.event.MouseEvent
evt) {
jButton1.setForeground(Color.white);}
private void jButton2MouseExited(java.awt.event.MouseEvent
evt) {
jButton2.setForeground(new Color(0,50,73)); }
private void jButton2MouseMoved(java.awt.event.MouseEvent
evt) {
jButton2.setForeground(Color.white);  }
```

- STACK DS :

```
private void
jButton1ActionPerformed(java.awt.event.ActionEvent evt)
{k=Integer.parseInt(ele.getText());
ele.setText("");
popp.setText("");
if(index>=9)
JOptionPane.showMessageDialog(null,"STACK is Full!");
else
{ index++;
stack.get(index).setText(""+k);
for(int i=0;i<10;i++)
stack.get(i).setBackground(new Color(154,209,212));
stack.get(index).setBackground(new Color(154,159,209));}}

private void jButton2ActionPerformed(java.awt.event.ActionEvent
evt)
{if(index<0)
JOptionPane.showMessageDialog(null,"STACK is empty!");
Else
{popp.setText(stack.get(index).getText());
stack.get(index).setText("");
index--;
for(int i=0;i<10;i++)
stack.get(i).setBackground(new Color(154,209,212));
if(index!=-1)
stack.get(index).setBackground(new Color(154,159,209));}}
```

- QUEUE DS:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {
if((R+1)>9)
 JOptionPane.showMessageDialog(null,"Queue is Full!");
else{ if(L==-1 && R==-1)
{R=0;L=0;}
else
R++;
q.get(R).setText(d.getText());}
d.setText("");    }
private void jButton3ActionPerformed(java.awt.event.ActionEvent
evt)
{t.setText("");
if(R<L || L==-1)
{JOptionPane.showMessageDialog(null,"Queue is Empty!");
 R=-1;
L=-1;   }
else
{t.setText(q.get(L).getText());
q.get(L).setText("");
L++;}
```

- TREE DS :

```
void Inorder(Node node)
{if(node==null)
return;
Inorder(node.left);
s.push(node.key);
for(int k=0;k<15;k++)
if((""+node.key).equals(io.get(k).getText()))
o.push(k);
Inorder(node.right);}
void Postorder(Node node){
if(node==null)
return;
Postorder(node.left);
Postorder(node.right);
s.push(node.key);
for(int k=0;k<15;k++)
if((""+node.key).equals(io.get(k).getText()))
o.push(k);}
void Preorder(Node node)
{if(node==null)
return;
s.push(node.key);
for(int k=0;k<15;k++)
if((""+node.key).equals(io.get(k).getText()))
o.push(k);
Preorder(node.left);
Preorder(node.right);}
```

- BUBBLE SORT:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {
if(j<n-1)
{int k=0;int flg=0;
for (int i = 0; i < n-j-1; i++)
{int a=Integer.parseInt(T.get(i).getText());
int b=Integer.parseInt(T.get(i+1).getText());
if(a>b)
{String t=T.get(i+1).getText();
T.get(i+1).setText(T.get(i).getText());
T.get(i).setText(t);
k=i+1;
flg=1;}}
T.get(k).setBackground(Color.red);
if(flg==1)
l++;
if(j==n-2)
l++;
passno.setText(""+(l));}
else
{JOptionPane.showMessageDialog(null,"DONE!!");}
j++;}
```

- SELECTION SORT :

```
private void
jButton3ActionPerformed(java.awt.event.ActionEvent evt) {
n=Integer.parseInt(N.getText());
if(n>10)
JOptionPane.showMessageDialog(null,"Please enter a number less
than 10!");
Else
{for(int i=0;i<n;i++)
{T.get(i).setBackground(Color.white);
T.get(i).setBackground(Color.cyan);
T.get(i).setEditable(true);}
N.setEditable(false);}}
private void jButton2ActionPerformed(java.awt.event.ActionEvent
evt) {
for(int i=0;i<n;i++)
T.get(i).setBackground(Color.white);
if(j<n)
{int min =j;passno.setText(""+(j+1));
for(int k=j;k<n;k++)
{int a=Integer.parseInt(T.get(k).getText());
int b;
b = Integer.parseInt(T.get(min).getText());
if(a<b)
min=k;}
swap(T.get(min),T.get(j));}
else
JOptionPane.showMessageDialog(null,"DONE!!");
j++;}
```

- INSERTION SORT:

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {
if(j<n)
{passno.setText(""+j);
String key=T.get(j).getText();
int i=j-1;
while
(i>=0 && Integer.parseInt(key)<Integer.parseInt(T.get(i).getText()))
{T.get(i+1).setText(T.get(i).getText());
i--;}
T.get(i+1).setText(key);
T.get(j).setBackground(new Color(137,218,89));}
else
JOptionPane.showMessageDialog(null,"DONE !!")
;j++;}
```

- BINARY SEARCH :

```
private void jButton1ActionPerformed(java.awt.event.ActionEvent
evt) {
if(low<high)
{int mid=(low+high)/2;
for(int i=0;i<n;i++)
T.get(i).setBackground(Color.white);
T.get(mid).setBackground(Color.gray);
if(Integer.parseInt(T.get(mid).getText())==key)
{ JOptionPane.showMessageDialog(null,"Element found at location
:"+(mid+1));
jButton1.setEnabled(false);
rslt.setText("FOUND");
loc.setText(""+(mid+1));}
else if(Integer.parseInt(T.get(mid).getText()) < key)
low=mid+1;
else
high=mid-1;
else
{JOptionPane.showMessageDialog(null,"Element not found");
rslt.setText("NOT FOUND");}}
```

- SCREEN SHOTS:

Fig . 2.2

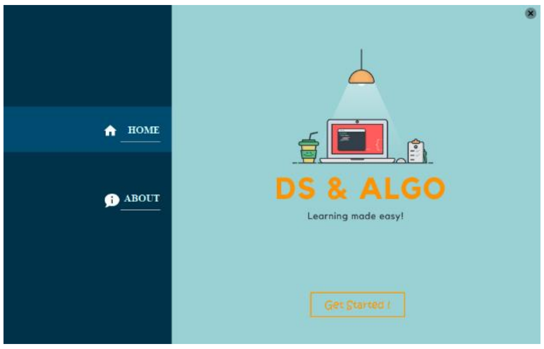
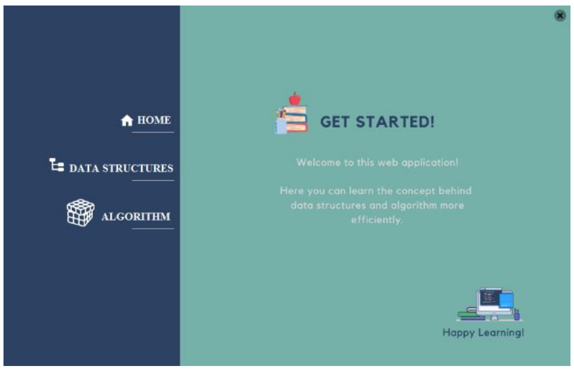


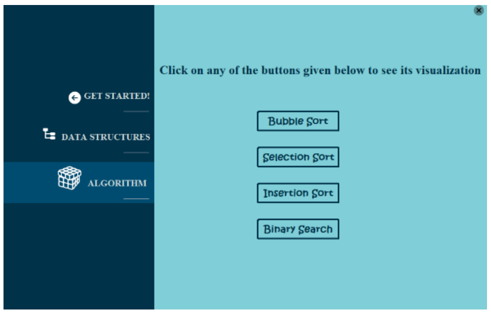
Fig. 2.3



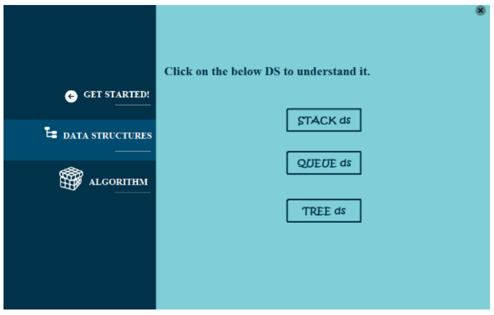
Welcome to this web application!

Here you can learn the concept behind data structures and algorithm more efficiently.

Fig. 2.4



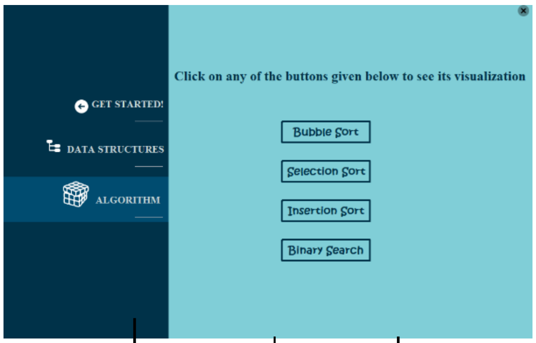
Click on any of the buttons given below to see its visualization



Click on the below DS to understand it.

Fig. 2.5

Fig. 2.6



Click on any of the buttons given below to see its visualization

Fig. 2.7

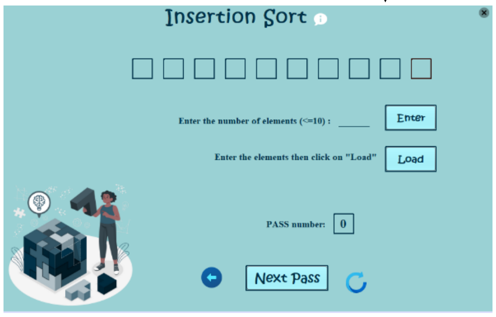

Insertion Sort
Enter the number of elements (<=10) :
Enter
Enter the elements then click on "Load"
Load
PASS number:
0
Next Pass

Fig. 2.8

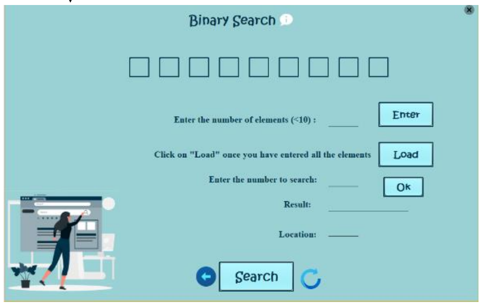

Binary Search
Enter the number of elements (<10) :
Enter
Click on "Load" once you have entered all the elements
Load
Enter the number to search:
Ok
Result:
Location:
Search

Fig. 2.9

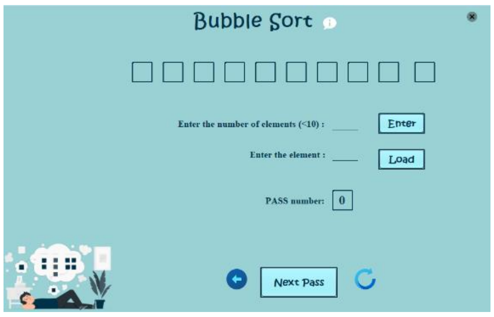


Fig. 2.10

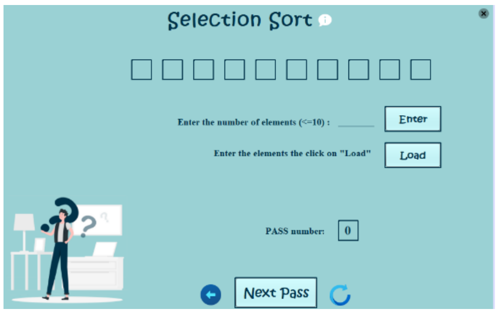
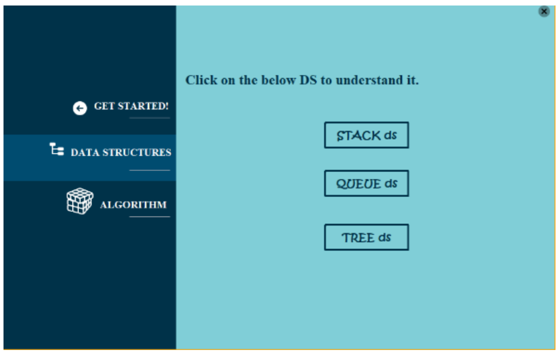

Selection Sort
Enter the number of elements (<=10) :
Enter
Enter the elements the click on "Load"
Load
PASS number:
0
Next Pass

Fig. 2.11



Click on the below DS to understand it.

Fig. 2.12

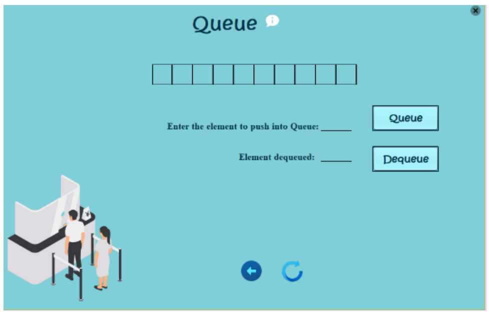


Fig.2.13

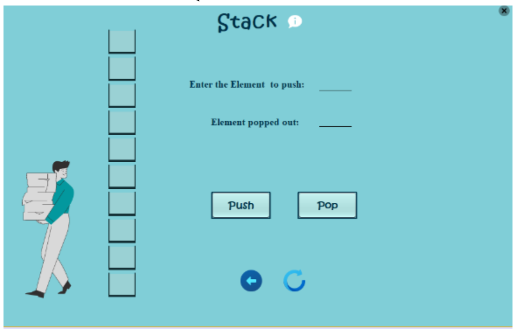


Fig.2.14

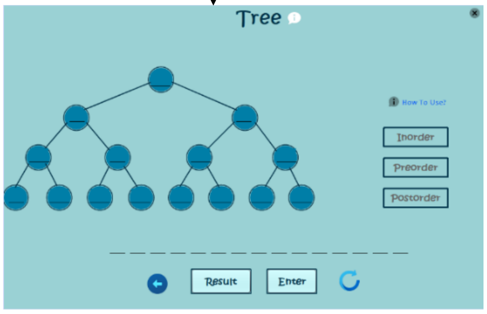
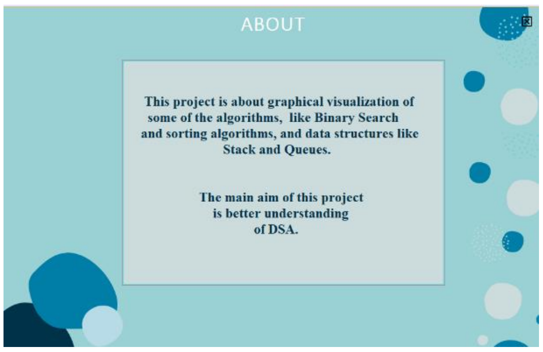


Fig.2.15



This project is about graphical visualization of some of the algorithms, like Binary Search and sorting algorithms, and data structures like Stack and Queues.

The main aim of this project is better understanding of DSA.

Fig.2.16

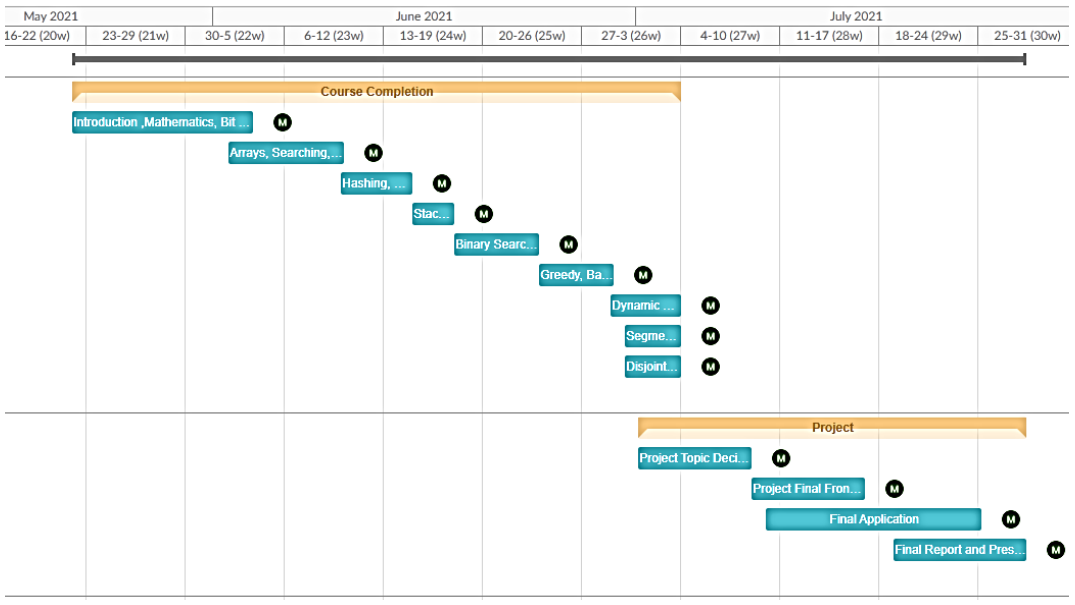
CHAPTER 3

CONCLUSION

LEARNING OUTCOME

Through this internship, that is MOOC, I was able to revise all the previous concepts and learn new concepts of Data Structures and Algorithm. Also, I was able to make a mini – project using Java Swing.

GANTT CHART


May 2021
June 2021
July 2021
16-22 (20w)
23-29 (21w)
30-5 (22w)
6-12 (23w)
13-19 (24w)
20-26 (25w)
27-3 (26w)
4-10 (27w)
11-17 (28w)
18-24 (29w)
25-31 (30w)
Course Completion
Introduction ,Mathematics, Bit ...
Arrays, Searching,...
Hashing, ...
Stac...
Binary Searc...
Greedy, Ba...
Dynamic ...
Segme...
Disjoint...
Project
Project Topic Deci...
Project Final Fron...
Final Application
Final Report and Pres...
M

PROJECT LEGACY

Through this summer MOOC , I came to know about lot of new concepts and new technologies. Also I became more efficient in time management. This project has also helped me to get better in Java Swing, about which I was not much aware in the past.

REFERENCES

- https://www.javatpoint.com/data-structure-tutorial
- https://www.javatpoint.com/data-structure-algorithm
- https://www.geeksforgeeks.org/

All the websites were accessed frequently almost every day during MOOC completion