

A REPORT ON PRACTICAL ASSIGNMENT OF COMPUTER GRAPHICS

Submitted By Ram Jonchhen

September 5, 2020

Contents

1	INTRODUCTION	2
1.1	BACKGROUND	2
1.1.1	BACKGROUND OF THE SORTING ALGORITHM	2
1.1.2	BACKGROUND OF THE PROJECT	2
1.2	STATEMENTS OF PROBLEMS	2
1.3	OBJECTIVE OF THE STUDY	3
2	LITERATURE REVIEW	4
2.1	HISTORY OF SORTING ALGORITHM	4
2.2	REVIEW OF THE TOOLS USED	4
2.3	HISTORY OF SIMILAR OTHER PROJECTS	6
3	RESEARCH AND METHODOLOGY	7
3.1	BUILDING BLOCKS OF THE PROJECT	7
3.1.1	DISPLAYING THE RECTANGULAR BLOCKS	7
3.2	ANIMATIONS / TRANSITIONS / DECORATIONS	8
3.3	ACTIONS ON ALGORITHMS	8
3.4	DIFFERENT WINDOWS	8
3.5	HOMEPAGE INTERFACE FOR THE USER	8
4	RESULTS AND DISCUSSION	9
5	PROJECT SCHEDULE	15
6	CONCLUSION AND RECOMMENDATION	16
6.1	CONCLUSION	16
6.2	LIMITATION OF THE STUDY	16
6.3	FUTURE ENHANCEMENTS	16

Chapter 1

INTRODUCTION

1.1 BACKGROUND

1.1.1 BACKGROUND OF THE SORTING ALGORITHM SORTING ALGORITHM

Sorting is the process of arranging the elements of a list or collection in increasing or decreasing order of same property. List should be homogenous i.e. all elements in the list should be of the same type. Those algorithms which do the process of sorting are called sorting algorithms.

Some importance's of sorting algorithms are as follows:

1. As humans are familiar to sorted data, sorting algorithms are used for producing human readable output.
2. Efficient sorting is important for optimizing the efficiency of other algorithms (such as merge and sort algorithms) that require input data to be in sorted lists.

1.1.2 BACKGROUND OF THE PROJECT

Visualization and animation helps us to learn and understand the concepts clearly. Sorting Algorithms when taught and while learning is theory based. The tracing of the process where every pass of the algorithms is redundant and paper based. In this technology age, why learn through the old way? The goal of the project is to help to understand the sorting algorithm through visuals and graphics effectively.

1.2 STATEMENTS OF PROBLEMS

While learning sorting algorithms and trying to understand what is going on in each step through dry run and tracing is difficult which makes understanding of the algorithm difficult.

The main goal of the project is to help the user to understand sorting algorithms clearly though visuals and animations, as visual and animations help us to understand topics more clearly and precisely.

While seeing other projects related to visualization of the sorting algorithms mostly the following things were found.

1. User can choose data sizes but not what data to insert to see the sorting process.
2. Animations are relatively fast and perform on very large number of data which makes understanding little bit difficult.
3. Animated videos of the sorting process were found, but as it is a video the process is shown in a pre-defined data size and data so user don't have the flexibility of selecting individual choice of data.

Project's goal is also to solve these problems as:

1. A platform/interface is given where user can select desired data size and data to sort, and sorting algorithm according to user size.
2. Clear and Understandable animations are shown in each sorting process in slow manner which helps in understanding sorting algorithm.

1.3 OBJECTIVE OF THE STUDY

The main objectives of the study are as follows:

1. To understand and learn how to use tools related to computer graphics and to implement them to create projects.
2. To learn the graphics tools provided by c graphics library and understanding how to use them and process them in Code::Blocks IDE.
3. To know about sorting algorithms and how to present them in understanding manner.

Chapter 2

LITERATURE REVIEW

2.1 HISTORY OF SORTING ALGORITHM

From the beginning of computing, the sorting problem has attracted a great deal of research, perhaps due to the complexity of solving it efficiently despite its simple, familiar statement. Among the authors of early sorting algorithms around 1951 was Betty Holberton (born Snyder), who worked on ENIAC and UNIVAC. Bubble sort was analyzed as early as 1956.

Comparison sorting algorithms have a fundamental requirement of $\Omega(n \log n)$ comparisons (some input sequences will require a multiple of $n \log n$ comparisons); algorithms not based on comparisons, such as counting sort, can have better performance. Asymptotically optimal algorithms have been known since the mid-20th century—useful new algorithms are still being invented, with the now widely used Timsort dating to 2002, and the library sort being first published in 2006.

2.2 REVIEW OF THE TOOLS USED

PROGRAMMING LANGUAGE USED

Programming language used for this project is C++ programming language. Functional approach of programming is used. As it is a visual project, graphics.h library is used for creating the graphics and visuals. And Code::Blocks ide is used as a building platform.

Following are the details of the tools, libraries used throughout the making of the project.

1. Code::Blocs IDE

Code::Blocks is a free C, C++ and Fortran IDE. It is built around plugin framework through which Code::Blocks can be extended with plugins. Extra functionality can be added by installing/coding a plugin.

2. Graphics.h header file

The graphics.h header file provides access to a simple graphics library that makes it possible to draw lines, rectangles, ovals, arcs, polygons, images, and strings on a graphical window.

Some of the functions used of the graphics.h header file are as follows:

Function used	Description
Line(int x1,int y1,int x2,int y2)	Creates a line from point (x1,y1) to (x2,y2).
Rectangle(int Xleft,int Yleft,int Xright,int Yright))	Creates a rectangle with top left corner (Xleft,Yleft) and bottom right corner (Xright,Yright) .
Cleardevice()	The header file graphics.h contains cleardevice() function which clears the screen in graphics mode and sets the current position to (0,0). Clearing the screen consists of filling the screen with current background color.
Setfillstyle(int pattern, int color)	It is used for setting the current fill pattern and color for floodfill(). Pattern value ranges from 0 to 12 varying from solid fill, line fill etc. Color ranges form 0 to 15 for various color.
Floodfill(int x,int y, int bordercolor)	It flood fills from point (x,y) in the pattern and color specified by setfillstyle().
Outtextxy(int x, int y, char *string) where, x, y are coordinates of the point and, third argument contains the address of string to be displayed.	It displays text or string at given coordinate form the character array or string give in the function.

Some other functions of other header files used in the projects are:

1. **Delay()function** It is used to suspend execution of a program for a particular time.

Syntax: Delay(unsigned int)

unsigned int is the number of milliseconds. The function is included under dos.h header file.

This is function is used to delay the program in order for the visuals to work and see as an animation.

2. **Sprintf()function**

Sprint() is used for storing the output in character buffer.

Syntax: sprint(char* ch,const char* string,...)

Example:

```
Char ch[50];
```

```
Int a=5,b=10,c;
```

```
C = a+b;
```

Sprint(ch,"Sum of Ch will contain: "sum of 5 and 10 is 15"; Sprint() is used in order to display the numbers in the visuals as outtextxy() takes fixed string.And numbers stored in variables directly can't be displayed through outtextxy().As outtextxy(char*) takes input parameter as character pointer, first the number stored in variables are stored in the character pointer through sprint() and displayed by outtextxy() by displaying the same character pointer.

2.3 HISTORY OF SIMILAR OTHER PROJECTS

While seeing other projects, some similar projects were found and also animated videos of sorting algorithms were found. Here are list of it.

1. **Visualgo.net**

url: <https://visualgo.net/en>

These platform provided visualization of sorting algorithms along with many other algorithms.

2. **Sorting Algorithm Animation**

url: <https://github.com/vbohush/SortingAlgorithmAnimations>

These project found was written in java. And showed visualization of many sorting algorithms like: bubble sort, comb sort, cocktail sort, etc.

3. **Animated Video**

url: <https://www.youtube.com/playlist?list=PL2aHrV9pFqNS79ZKnGLw-RG5gH01bcjRZ>

Some animated videos was found where the sorting algorithm were explained with great understanding.

Chapter 3

RESEARCH AND METHODOLOGY

Following were the approaches made while building this project.

3.1 BUILDING BLOCKS OF THE PROJECT

As we are visualizing sorting algorithms I took the following decisions.

1. For the process of sorting, numbers was decided to be sorted. For visual representation of the numbers, I decided to display numbers in the form of rectangular blocks
2. Among the sorting algorithms bubble sort and selection sort were chosen.

3.1.1 DISPLAYING THE RECTANGULAR BLOCKS

As this is a visualization project, displaying the visuals is the main important part of the program. This part of project is the most crucial part of the project. Displaying of the rectangular blocks was achieved by following means.

1. TAKING INPUT

First of all to display the numbers in the form of rectangular blocks input from the user is required, for storing the user input array data structure is used. User input is taken as:

- (a) User is first made to enter the no of input to display. (current the maximum no of input is 10).
- (b) User is then made to input the data.

This is achieved in the program through the *take_input()* function in the program.

2. DISPLAYING FUNCTION

This part of the program is the backbone of the project. In the program displaying of rectangular blocks is done through the *display_blocks()* function. It simply takes array as it's argument and displays the rectangular blocks according to the content in the array. It displays the values with colorful rectangle with the respective value in it.

3.2 ANIMATIONS / TRANSITIONS / DECORATIONS

Animations and transitions are necessary for this project. For explaining various algorithms, various animations are used. Animations like two blocks moving up for swapping of two blocks were used, for insertion sort scanning minimum element for each block was shown by moving each blocks downward. Similarly Decorations like balancer between two compared numbers can be seen in the figure above. Other decorations like various texts like “numbers can be swapped” is shown if number can’t be swapped in the bubble sort, “Min” is shown in the element found minimum in the selection sort, “Sorting is completed” is shown after the sorting process is completed. Also delay() function is used to show the animations in a effective manner.

3.3 ACTIONS ON ALGORITHMS

For the sorting algorithms to be visualized the process of the algorithms should be traced step by step and visualized too. Every steps of the algorithms is made call to the various functions in the programs through which the step is clearly explained.

For ex: in bubble sort comparison

If(arr[i]>arr[j+1])

Then show swapping of the two numbers animations

Else

Show the text “Numbers can’t be swapped”

Likewise in every part of the sorting algorithms for visual representation of the steps various functions are used for explaining the algorithm

3.4 DIFFERENT WINDOWS

User interface for choosing the algorithm and giving the input is done from a window whereas the visualization of the algorithms is shown in another window.

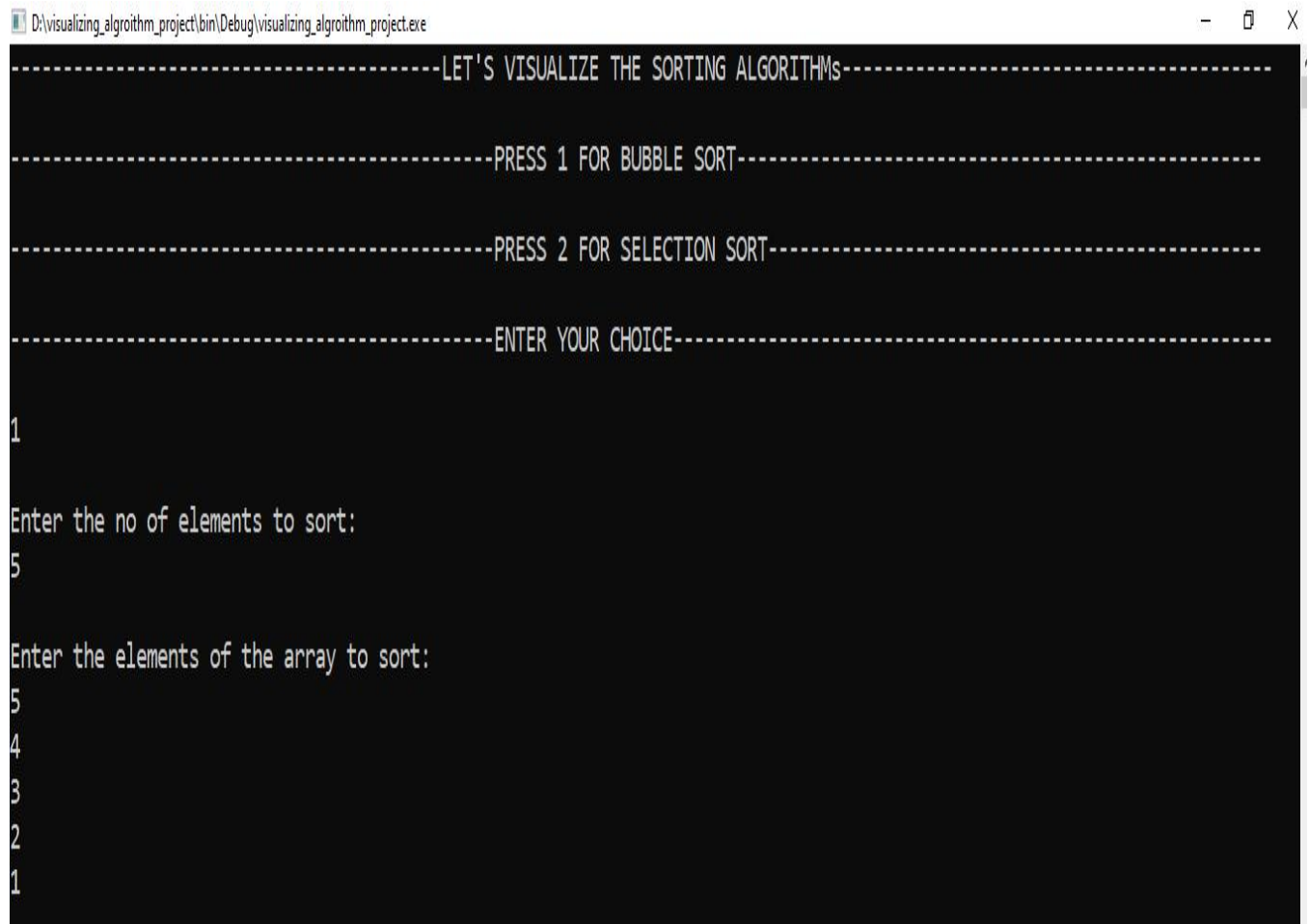
3.5 HOMEPAGE INTERFACE FOR THE USER

For user to use the program and for seeing the visuals of the algorithms a homepage was created where user can select desired algorithm bubble/ selection sort and give input and see the visualization of the algorithm.

Chapter 4

RESULTS AND DISCUSSION

Following results of the projects were obtained:



```
D:\visualizing_algorithm_project\bin\Debug\visualizing_algorithm_project.exe
-----LET'S VISUALIZE THE SORTING ALGORITHMs-----
-----PRESS 1 FOR BUBBLE SORT-----
-----PRESS 2 FOR SELECTION SORT-----
-----ENTER YOUR CHOICE-----
1
Enter the no of elements to sort:
5
Enter the elements of the array to sort:
5
4
3
2
1
```

Figure 4.1: A home screen where users can choose the sorting algorithm and give input was made:

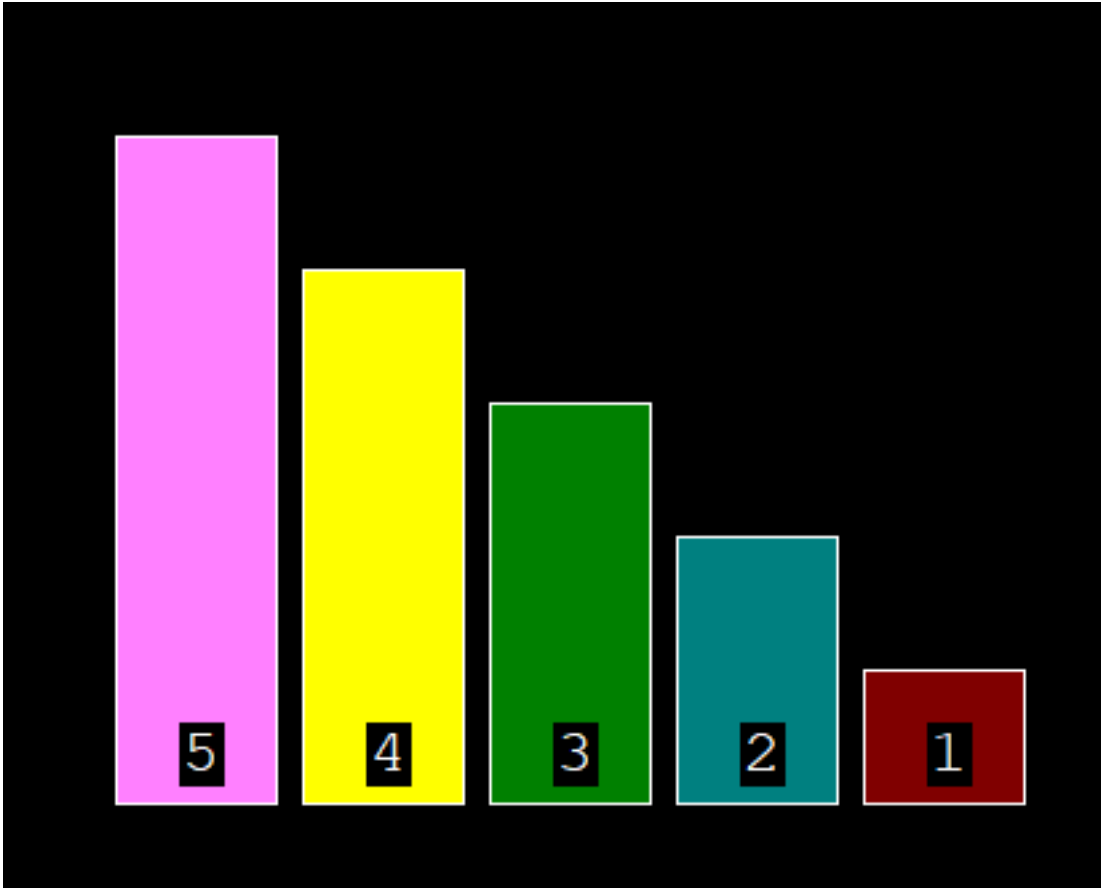


Figure 4.2: Numbers were displayed in the form of colored rectangular blocks with numbers in it.

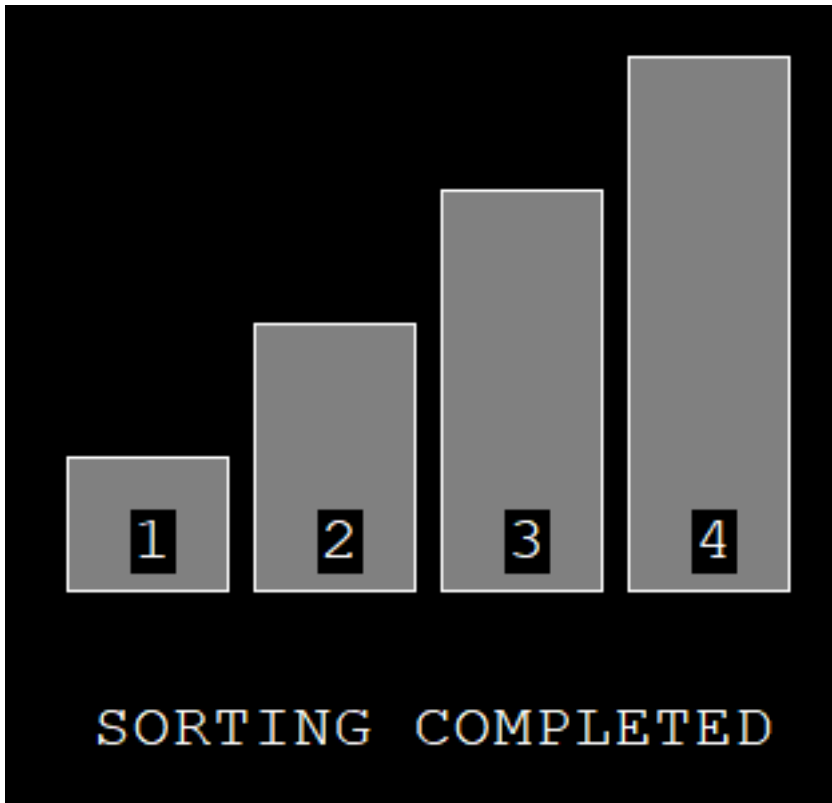


Figure 4.3: After the sorting process was completed the message of “sorting completed” was shown.

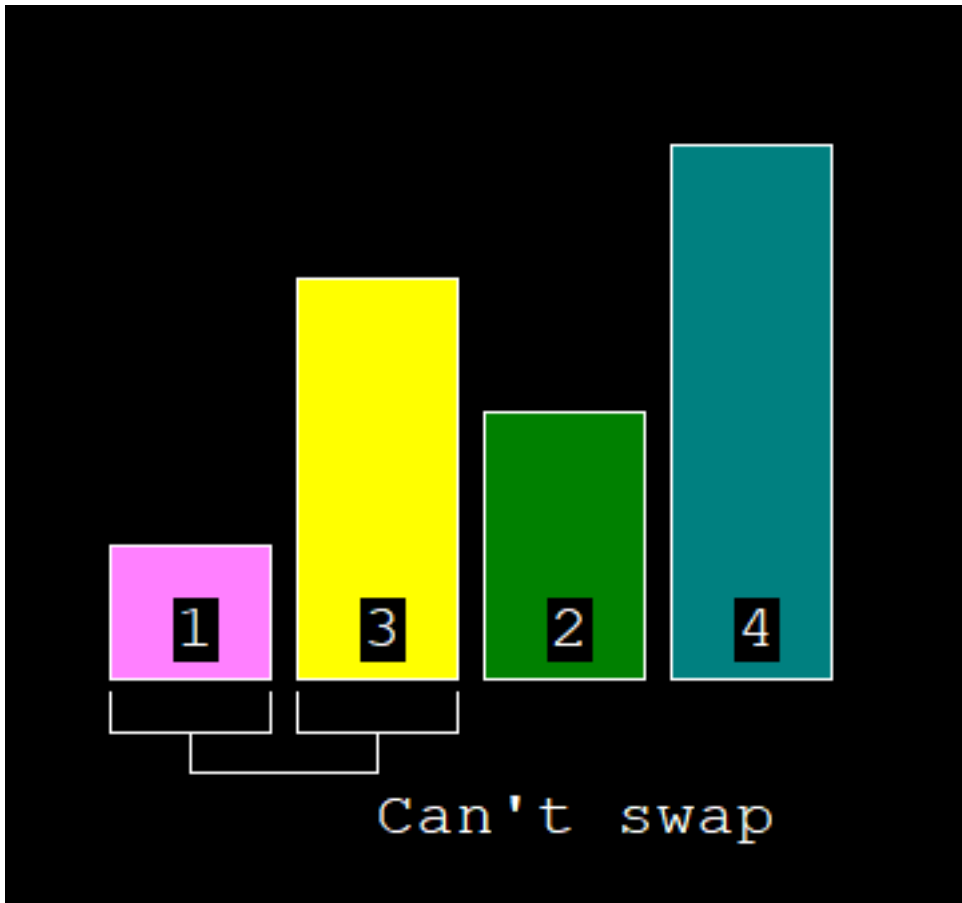


Figure 4.4: A balancer was used to compare two numbers and if numbers can be swapped animation of rectangular blocks was shown else a message of “can’t swap” was shown.

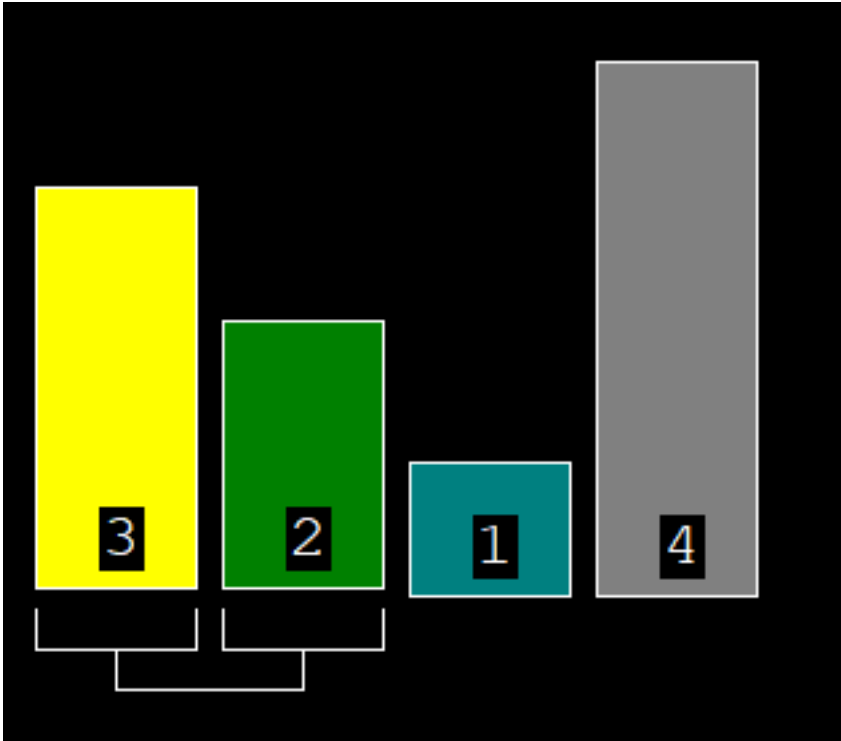


Figure 4.5: Blocks moving up animation for swapping of numbers in bubble sort

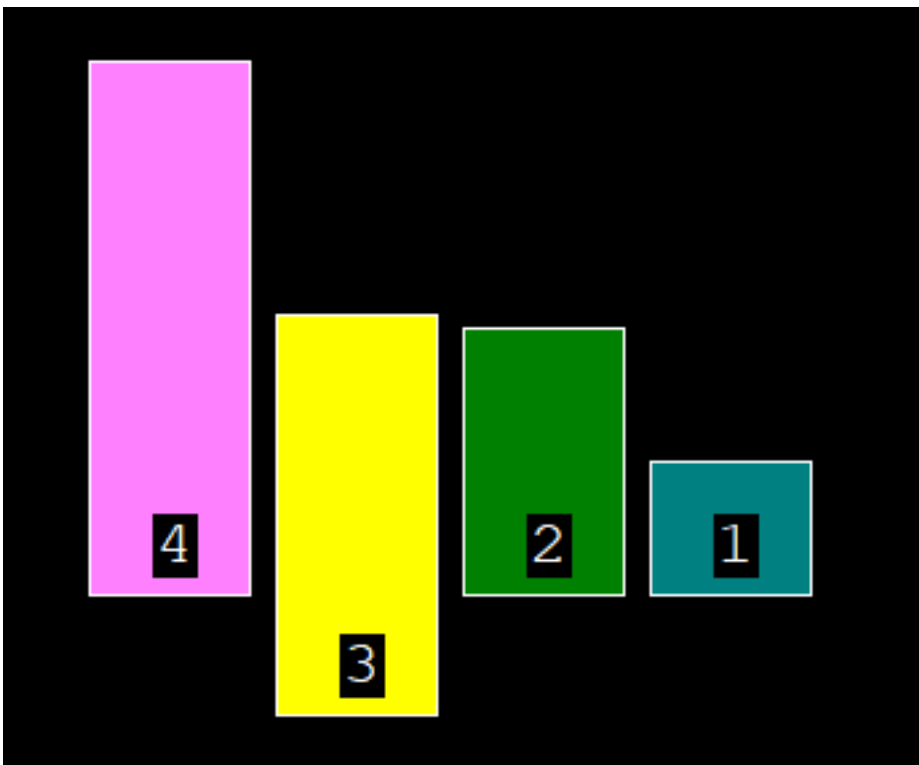


Figure 4.6: For visuals of scanning for minimum element in the selection sort rectangular blocks moving down animation was shown.

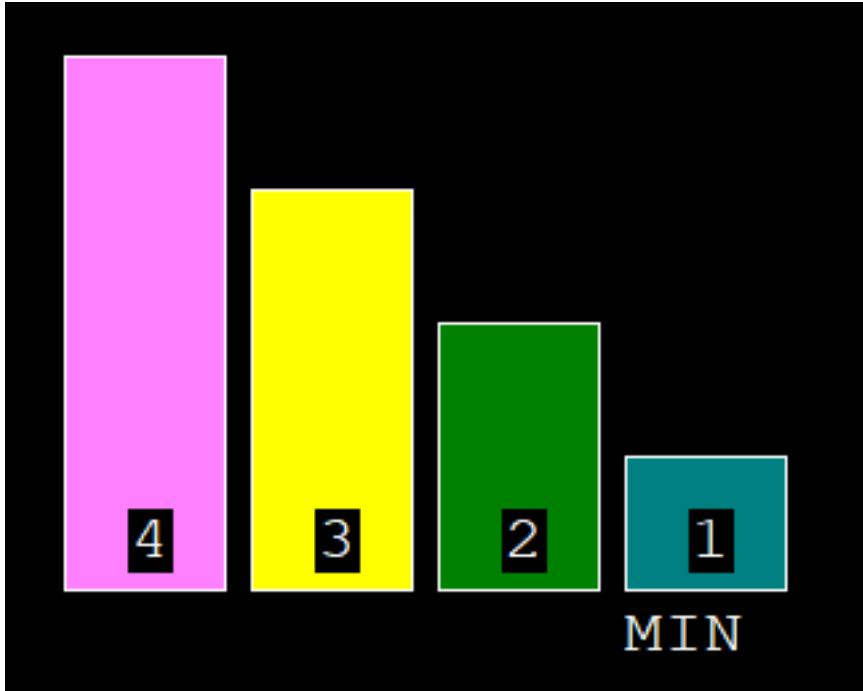


Figure 4.7: Below the minimum element found “MIN” text was shown

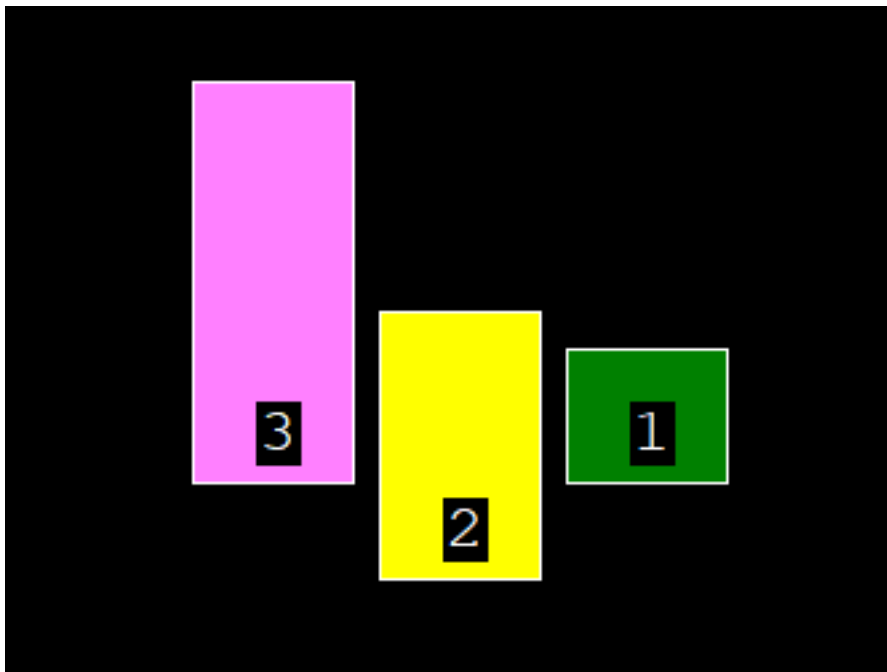


Figure 4.8: swapping of elements was shown by moving the two elements to swap in upward direction for selection sort.

Chapter 5

PROJECT SCHEDULE

The project development process can be seen in the table below:

Date	Details
Feb20 2020	Started with making of the project Basic Prototype and Code of Display function made.
Mar2 2020	Display Function code is done, Display function now display numbers on it on it's visual.
Mar3 2020	Basic Prototype of bubble sort and some of it's animation was developed.
Mar4 2020	Balancer and Text animation done for Bubble sort visualization was made.
Mar6 2020	Merge sort visualization was developed.
Mar7 2020	Homescreen for user interaction int the project was made.
Mar19 2020	Code refactoring was done and presentation of project was made.
Sep1 2020	Started with report writing of the project.
Sep5 2020	Finished with report writing of the project.

Chapter 6

CONCLUSION AND RECOMMENDATION

6.1 CONCLUSION

Concept and history of sorting algorithms was known and clearly understood. Tools like code blocks ide and using of graphics.h library was known for creating graphics project using c++ programming language.

A Platform where sorting algorithms can be visualized through user input was created. Platform consisted of visualization of bubble/selection sort where these algorithms were visualized with animations and transitions.

6.2 LIMITATION OF THE STUDY

- Currently users are only able to give fixed number of input for visualization (currently 10).
- Only two sorting algorithms are shown selection and bubble sort for visualization.
- When the no of inputs are high, displaying of the graphics becomes disturbing as the display in the screen blinks (flickering problem) which makes compromise on user experience.

6.3 FUTURE ENHANCEMENTS

- Flexibility in the input for the user can be improved by giving the user to input more amounts of data.
- More sorting algorithms can be introduced for visualization.
- Concept of double buffer can be introduced for removing flickering in the screen for higher input size and numbers.