SORTING ALGORITHM VISUALIZER

Neelu yadav[1], Amisha tripathi[2]

Department of Information Technology, Raj Kumar Goel Institute of Technology, Delhi NCR, Ghaziabad 201206, UP, India

Abstract

Algorithm visualizer is high-level is a high level dynamic visualizer of software which uses UI techniques to monitor the computational and portray steps of algorithms.Algorithm visualizer is a useful tool in algorithm engineering particularly at many stages of designing,implementing and experimental evaluation, and presentation of the algorithms.

Algorithm visualizer is used to sort the algorithms like bubble sort ,merge sort, heap sort etc.The main motive of the study is to design a system which can sort the algorithms and implements on systems.

Keywords: bubble sort,merge sort.

I. INTRODUCTION

The project is based on GUI application, It is a Sorting Algorithm Visualizer.The project helps to sort the algorithms such as bubble sort, merge sort etc. It is a graphical user interface application which is an interface or a window based appliaction or software which runs without the help of internet. The application generates an array and sorts the array according to the selected sorting algorithms.

Technology used: The technology used in the project are:

-PYTHON
- TKINTER

PYTHON: It is an interpreted high-level general purpose programming language.It supports modules and packages ,which encourages program modularity and code reuse.

TKINTER: It is python's standard GUI package. Out of all the GUI methods,Tkinter is the most commonly used method.Python with Tkinter is the fastest and easiest way to create the GUI applications.

Figure 1 .main window of the application

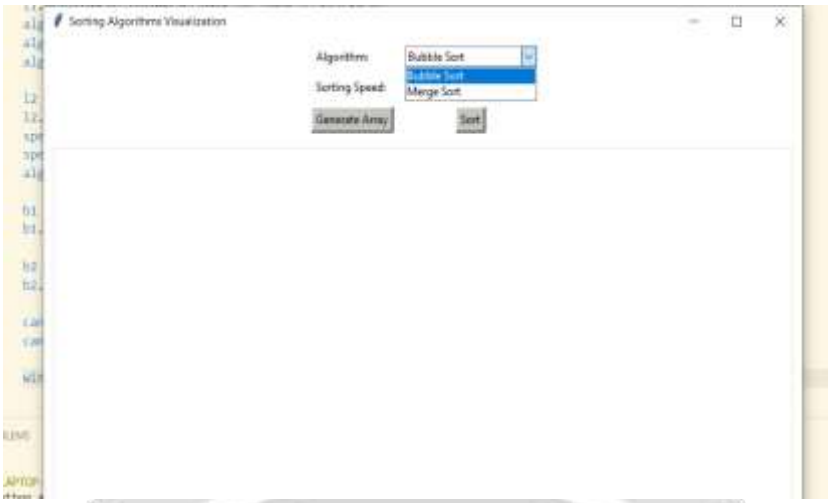

Figure 2. selection of sorting algorithm

. II. LITERATURE REIVEW

The starting research for the project was done by the working group at conference of ITiCSE 2002 (Napset al.. 2003a).

III. COMPONENTS

Components of the project are-

MAIN.PY -This file contains the creation of interface with the help of Tkinter module
.Formation of frame and canvas inside the window .
Formation of labels and buttons for the selection of Algorithms , sorting speed,generate array ,sorting array etc.
Functions for the generation of the array,drawing the bars corresponding to numbers into array , selection of the speed ,sorting the array.

COLORS.PY - This module contains the colors in the form of Hexadecimal values for the bars, buttons ,window etc.

BUBBLESORT.PY - This module contains the function"Bubble_sort()"for applying the bubble sorting to the given generated array.

MERGESORT.PY - This module contains the function "merge_sort()"for applying the merge sorting to the given generated array.

IV. METHODOLOGY

This project is designed to sort algorithms which investigates and visualizes the worst and best case for each implemented algorithm.

Design steps: Tkinter is very easy GUI library that can be used to visualize sorting algorithms.Here bubble sort and merge sort are used to sort the algorithms.

Firstly the code for sorting algorithm is written and designed once the code have been written and designed , they have been synthesized and python codes have been run, now the user can choose searching and sorting algorithms to find the time complexity of sorting algorithms.

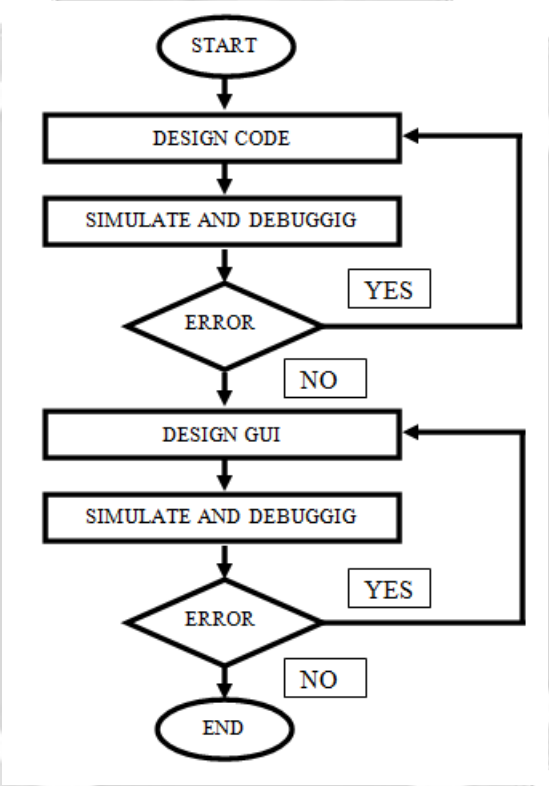

Figure 3. flowchart of system for algorithm visualizer

V.RESULT

When we select an algorithm for sorting an unsorted array is generated and we also choose the speed for sorting then this system selects the appropriate methods of sorting and sorts the given array and generates the output as shown in figure 4.

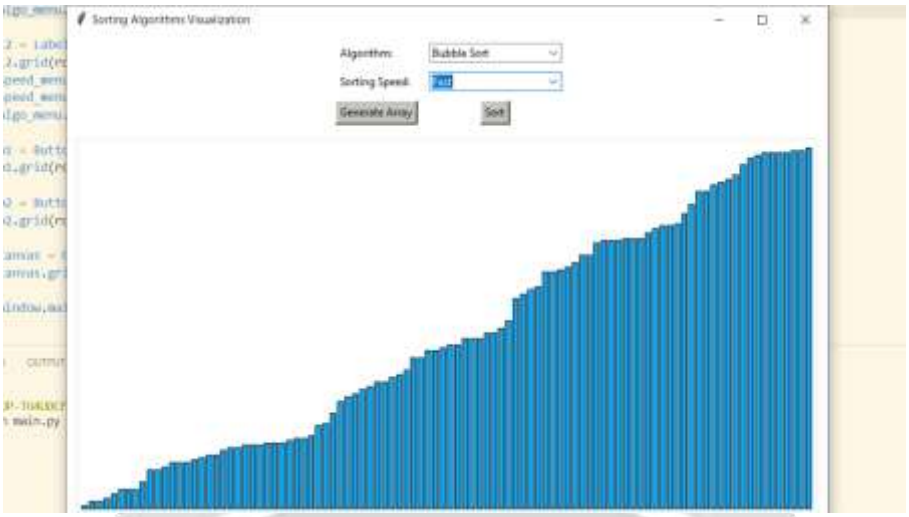

Figure 4. result of selected sorting algorithm

VI- CONCLUSIONS

The project is based on a GUI Application which is sorting algorithm visualizer which helps in sorting the given array by applying the several sorting methods.Algorithm visualizer can be a valuable tool which can be used in addition to standard ways of study in engineering field .Algorithm visualizer helps in understanding the principles.

REFERENCES

Ronald M. Baecker. Sorting Out Sorting: A Case Study of Software Visualization for Teaching Computer Science, chapter 24, pages 369–381. The MIT Press, Cambridge, MA, 1998.

Benjamin S. Bloom. Taxonomy of Educational Objectives, Handbook 1: Cognitive Domain. Addison Wesley, 1956.

John Domingue. Software visualization and education. In Stephan Diehl, editor, Software Visualization: International Seminar, pages 205–212, Dagstuhl, Germany, 2002. Springer.

Richard M. Felder. Matters of style. ASEE Prism, 6(4):18–23, 1996.

Jyrki Haajanen, Mikael Pesonius, Erkki Sutinen, Jorma Tarhio, Tommi Ter¨asvirta, and Pekka Vanninen. Animation of user algorithms on the Web. In Proceedings of Symposium on Visual Languages, pages 360–367, Isle of Capri, Italy, 1997. IEEE.

Christopher D. Hundhausen, Sarah A. Douglas, and John T. Stasko. A meta-study of algorithm visualization effffectiveness. Journal of Visual Languages & Computing, 13(3):259–290, June 2002

.

Ville Karavirta, Ari Korhonen, Jussi Nikander, and Petri Tenhunen. Efffortless creation of algorithm visualization. In Proceedings of the Second Annual Finnish / Baltic Sea Conference on Computer Science Education, pages 52–56, October 2002.

David A. Kolb, editor. Experiential Learning: Experience as the Source of Learning and Development. Prentice-Hall Inc, New Jersey, USA, 1984.

Ari Korhonen and Lauri Malmi. Matrix — Concept animation and algorithm simulation system. In Proceedings of the Working Conference on Advanced Visual Interfaces, pages 109–114, Trento, Italy, May 2002. ACM.

Paul LaFollette, James Korsh, and Raghvinder Sangwan. A visual interface for effffortless animation of C/C++ programs. Journal of Visual Languages and Computing, 11(1):27–48, 2000.

Fernando Naharro-Berrocal, Cristobal Pareja-Flores, Jaime Urquiza-Fuentes, J. Angel Vel´azquezIturbide, and Francisco Gort´azar-Bellas. Redesigning the animation capabilities of a functional programming environment under an education al framework. In Second Program Visualization Workshop, pages 59–68, HornstrupCentret, Denmark, 2002.

Thomas L. Naps, Guido R¨oßling, Vicki Almstrum, Wanda Dann, Rudolf Fleischer, Chris Hundhausen, Ari Korhonen, Lauri Malmi, Myles McNally, Susan Rodgers, and J. Angel Vel´azquez-Iturbide. Exploring the role of visualization and engagement in computer science education. SIGCSE Bulletin, 35(2):131–152, June 2003a.

Thomas L. Naps, Guido R¨oßling, Jay Anderson, Stephen Cooper, Wanda Dann, Rudolf Fleischer, Boris Koldehofe, Ari Korhonen, Marja Kuittinen, Charles Leska, Lauri Malmi, Myles McNally, Jarmo Rantakokko, and Rockford J. Ross. Evaluating the educational impact of visualization. SIGCSE Bulletin, 35(4):124–136, December 2003b.

Blaine A. Price, Ronald M. Baecker, and Ian S. Small. A principled taxonomy of software visualization. Journal of Visual Languages and Computing, 4(3):211–266, 1993.

Guido R¨oßling and Thomas L. Naps. A testbed for pedagogical requirements in algorithm visualizations. In Proceedings of the 7th Annual SIGCSE/SIGCUE Conference on Innovation and Technology in Computer Science Education, ITiCSE'02, pages 96–100, Aarhus, Denmark, 2002. ACM.

John T. Stasko. Using student-built algorithm animations as learning aids. In The Proceedings of the 28th SIGCSE Technical Symposium on Computer Science Education, pages 25–29, San Jose, CA, USA, 1997. ACM.