Sorting algorithm visualization : Heap Sort

Last Updated : 28 Dec, 2021

An algorithm like Heap sort can be understood easily by visualizing. In this article, a program that visualizes the Heap Sort Algorithm has been implemented.

The Graphical User Interface(GUI) is implemented in Python using pygame library.

Approach:

Generate random array and fill the pygame window with bars. Bars are straight vertical lines, which represents array elements.

- Set all bars to green color (unsorted).
- Heapify the array to perform sorting.
- After Heapify, large bars are at the beginning followed by smaller bars.
- Use pygame.time.delay() to slow down the algorithm, so that we can see the sorting process.
- Implement a timer to see how the algorithm performs.
- The actions are performed using 'pygame.event.get()' method, which stores all the events which user performs, such as start, reset.
- Blue color is used to highlight bars that are involved in sorting at a particular time.
- Orange color highlights the bars sorted.

Observations:

We can clearly see from the Heap Sort visualization, that Heap Sort is very fast compared to other sorting algorithms like Insertion sort or Selection sort and similar in speed with Merge sort.

Examples:

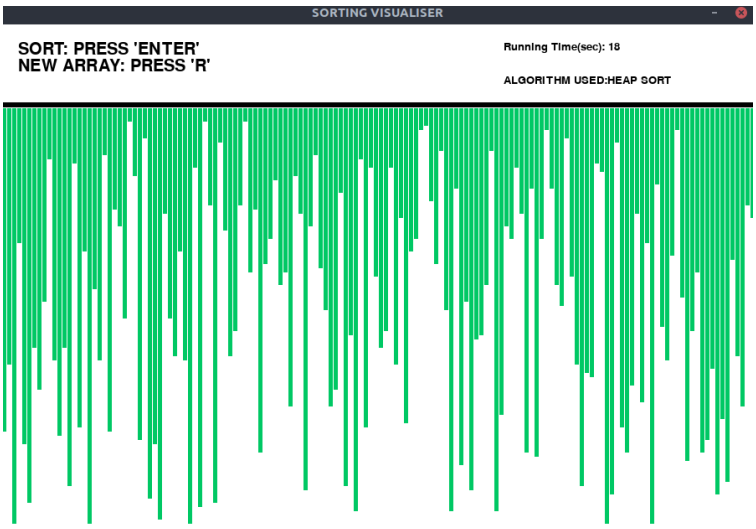
Input:

Press “Enter” key to Perform Visualization.

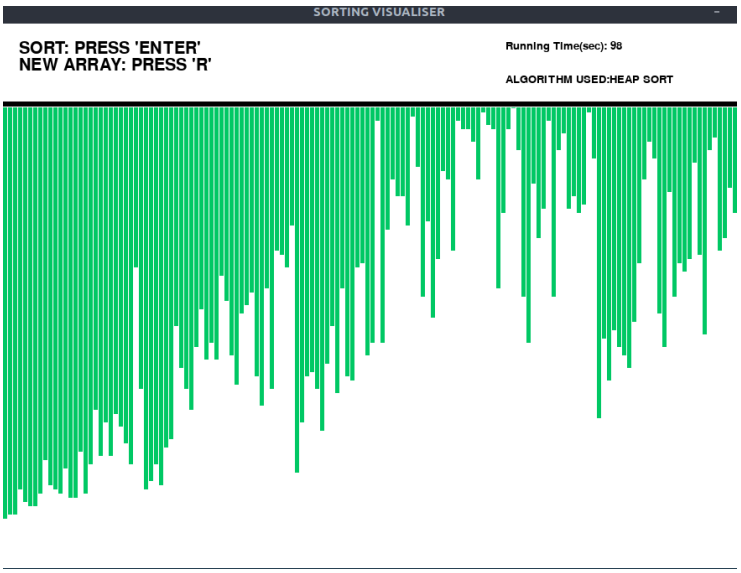
Press “R” key to generate new array.

Output:

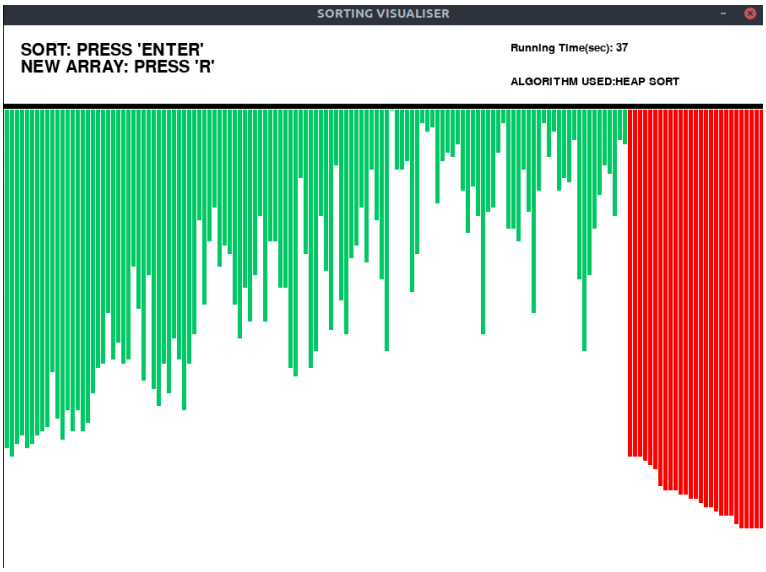
Initial:



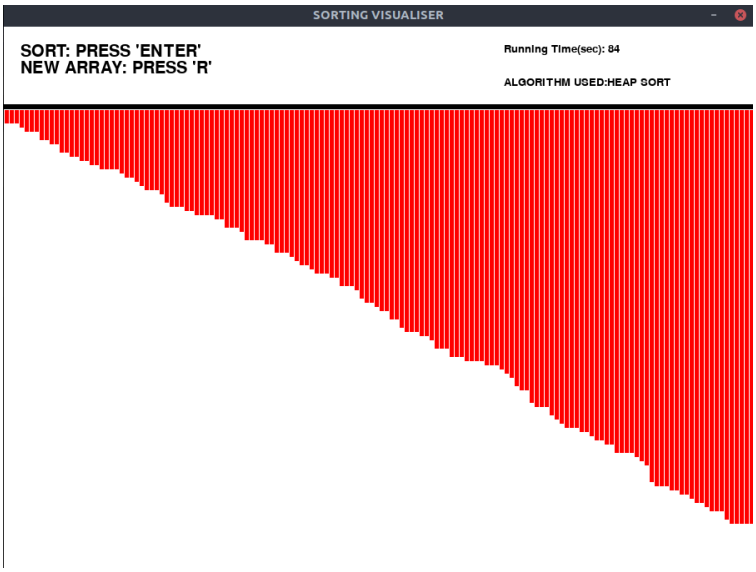
After heapification of array:



Sorting:



Final:



Please make sure to install the pygame library before running the below program.

Below is the implementation of the above visualizer:

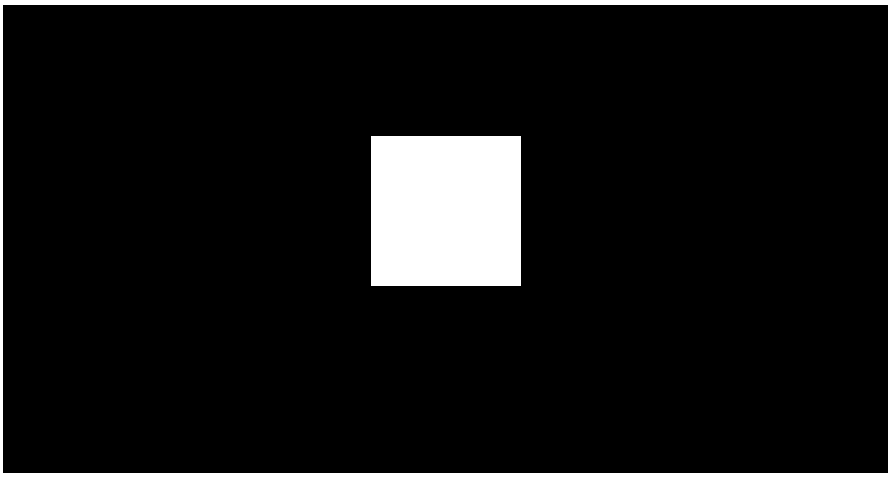
Python3

```
# Python implementation of the
# Sorting visualiser: Heap Sort

# Imports
```

```
import pygame
import random
import time
pygame.font.init()
startTime = time.time()

# Total window
screen = pygame.display.set_mode(
    (900, 650)
)

# Title and Icon
pygame.display.set_caption(
    "SORTING VISUALISER"
)

# Uncomment below lines for setting
# up the icon for the visuliser
# img = pygame.image.load('sorticon.png')
# pygame.display.set_icon(img)

# Boolean variable to run
# the program in while loop
run = True

# Window size and some initials
width = 900
length = 600
array = [0]*151
arr_clr = [(0, 204, 102)]*151
clr_ind = 0
clr = [(0, 204, 102), (255, 0, 0),
       (0, 0, 153), (255, 102, 0)]
fnt = pygame.font.SysFont("comicsans", 30)
fnt1 = pygame.font.SysFont("comicsans", 20)

# Function to generate new Array
def generate_arr():
    for i in range(1, 151):
        arr_clr[i] = clr[0]
        array[i] = random.randrange(1, 100)


# Initially generate a array
generate_arr()

# Function to refill the
# updates on the window
def refill():
```

```
        screen.fill((255, 255, 255))
        draw()
        pygame.display.update()
        pygame.time.delay(10)


    # Sorting Algorithm: Heap Sort
    def heapSort(array):
        n = len(array)
        for i in range(n//2-1, -1, -1):
            pygame.event.pump()
            heapify(array, i, n)
        for i in range(n-1, 0, -1):
            array[i], array[0] = array[0], array[i]
            arr_clr[i] = clr[1]
            refill()
            heapify(array, 0, i)


    def heapify(array, root, size):
        left = root * 2 + 1
        right = root * 2 + 2
        largest = root
        if left < size and array[left] > array[largest]:
            largest = left
        if right < size and array[right] > array[largest]:
            largest = right
        if largest != root:
            arr_clr[largest] = clr[2]
            arr_clr[root] = clr[2]
            array[largest],\
            array[root] = array[root],\
            array[largest]
            refill()
            arr_clr[largest] = clr[0]
            arr_clr[root] = clr[0]
            heapify(array, largest, size)
            refill()

    # Function to Draw the array values
    def draw():

        # Text should be rendered
        txt = fnt.render("SORT: PRESS 'ENTER'",
                         1, (0, 0, 0))
        # Position where text is placed
        screen.blit(txt, (20, 20))
        txt1 = fnt.render("NEW ARRAY: PRESS 'R'",
                          1, (0, 0, 0))
```

```
        screen.blit(txt1, (20, 40))
        txt2 = fnt1.render("ALGORITHM USED:" +
                           "HEAP SORT", 1, (0, 0, 0))
        screen.blit(txt2, (600, 60))
        text3 = fnt1.render("Running Time(sec): " +
                            str(int(time.time() - startTime)),
                            1, (0, 0, 0))
        screen.blit(text3, (600, 20))
        element_width = (width-150)//150
        boundry_arr = 900 / 150
        boundry_grp = 550 / 100
        pygame.draw.line(screen, (0, 0, 0), (0, 95),
                         (900, 95), 6)

        # Drawing the array values as lines
        for i in range(1, 151):
            pygame.draw.line(screen, arr_clr[i],
                             (boundry_arr * i-3, 100),
                             (boundry_arr * i-3,
                              array[i]*boundry_grp + 100),\
                             element_width)


    # Program should be run
    # continuously to keep the window open
    while run:
        # background
        screen.fill((255, 255, 255))

        # Event handler stores all event
        for event in pygame.event.get():

            # If we click Close button in window
            if event.type == pygame.QUIT:
                run = False
            if event.type == pygame.KEYDOWN:
                if event.key == pygame.K_r:
                    generate_arr()
                if event.key == pygame.K_RETURN:
                    heapSort(array)
        draw()
        pygame.display.update()

    pygame.quit()
```



00:00 00:28

Previous Article

Ternary Search Visualization using Pygame in Python

Next Article

Sorting algorithm visualization : Insertion Sort

Similar Reads

Difference between Binary Heap, Binomial Heap and Fibonacci Heap

Binary Heap:A Binary Heap is a Binary Tree with following properties. It's a complete binary tree i.e., all levels are completely filled except possibly the last...

2 min read

Sorting Algorithm Visualization : Merge Sort

The human brain can easily process visuals instead of long codes to understand the algorithms. In this article, a program that program visualizes the Merge sort...

3 min read

Sorting Algorithm Visualization : Quick Sort

An algorithm like Quicksort algorithm is hard to understand theoretically. We can understand easily by visualizing such kind of algorithms. In this article, a progra...

3 min read

Sorting algorithm visualization : Insertion Sort

An algorithm like Insertion Sort can be understood easily by visualizing. In this article, a program that visualizes the Insertion Sort Algorithm has been...

3 min read

What is the stupidest sorting algorithm? (Worst Sorting Algorithm)

Bogo sort stands out as the undisputed champion of stupidity. Unlike other sorting algorithms that follow a structured approach, Bogo sort relies on sheer...

2 min read

Heap Sort Visualization using JavaScript

GUI(Graphical User Interface) helps in better understanding than programs. In this article, we will visualize Heap Sort using JavaScript. We will see how the...

4 min read

Heap Sort for decreasing order using min heap

Given an array of elements, sort the array in decreasing order using min heap. Examples: Input : arr[] = {5, 3, 10, 1} Output : arr[] = {10, 5, 3, 1} Input : arr[] = {1...

13 min read

Sorting Algorithms Visualization | Selection Sort

The human brain can easily process visuals in spite of long codes to understand the algorithms. In this article, Selection Sort visualization has been implemented...

6 min read

Sorting objects using In-Place sorting algorithm

Given an array of red, blue and yellow objects, the task is to use an in-place sorting algorithm to sort the array in such a way that all the blue objects appear...

7 min read

Know Your Sorting Algorithm | Set 1 (Sorting Weapons used by Programmi…

Ever wondered how sort() function we use in C++/Java or sorted() in Python work internally? Here is a list of all the inbuilt sorting algorithms of different...

2 min read

Article Tags : Algorithms Analysis of Algorithms DSA Heap +8 More

Practice Tags : Algorithms Heap python Sorting

DSA for Beginners
Basic DSA Problems
DSA Roadmap
Top 100 DSA Interview Problems
DSA Roadmap by Sandeep Jain
All Cheat Sheets

Machine Learning
ML Maths
Data Visualisation
Pandas
NumPy
NLP
Deep Learning

Web Technologies

HTML
CSS
JavaScript
TypeScript
ReactJS
NextJS
Bootstrap
Web Design

Python Tutorial

Python Programming Examples
Python Projects
Python Tkinter
Web Scraping
OpenCV Tutorial
Python Interview Question
Django

Computer Science

Operating Systems
Computer Network
Database Management System
Software Engineering
Digital Logic Design
Engineering Maths
Software Development
Software Testing

DevOps

Git
Linux
AWS
Docker
Kubernetes
Azure
GCP
DevOps Roadmap

System Design

High Level Design
Low Level Design
UML Diagrams
Interview Guide
Design Patterns
OOAD
System Design Bootcamp
Interview Questions

Inteview Preparation

Competitive Programming
Top DS or Algo for CP
Company-Wise Recruitment Process
Company-Wise Preparation
Aptitude Preparation
Puzzles

School Subjects

Mathematics
Physics
Chemistry
Biology
Social Science
English Grammar
Commerce
World GK

GeeksforGeeks Videos

DSA
Python
Java
C++
Web Development
Data Science
CS Subjects