Binary Search Visualization using Pygame in Python

Last Updated : 28 Jul, 2020

An algorithm like Binary Search can be understood easily by visualizing. In this article, a program that visualizes the Binary Search Algorithm has been implemented. The Graphical User Interface(GUI) is implemented in Python using pygame library.

Approach

Generate random array, sort it using any sorting algorithm, and fill the pygame window with bars. Bars are straight vertical lines, which represent array elements.

- Set all bars to green color.
- Use pygame.time.delay() to slow down the algorithm, so that we can see the searching 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 the user performs, such as start, reset.
- The blue color is used to highlight the bar equal to the key if found.
- Orange color highlights the left and right bars.

Below is the implementation of the above visualizer:

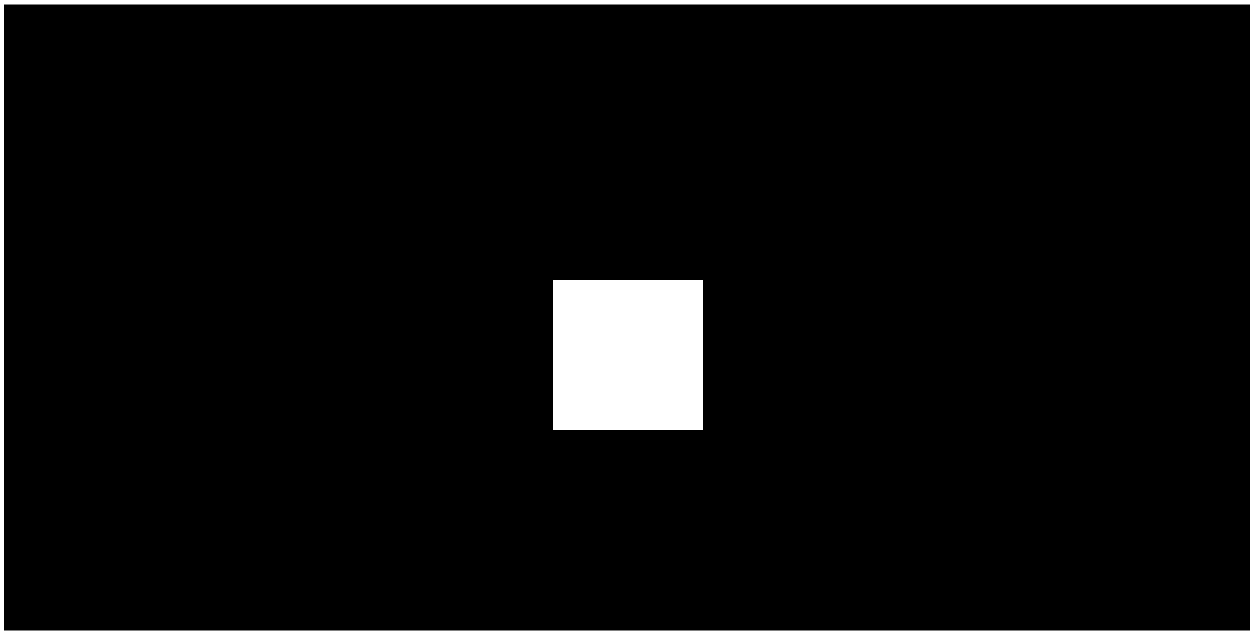
Python

```python
# Python implementation of the
# Sorting visualiser: Insertion 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(
    "BINARY SEARCH 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
key = 0
foundkey = False

arr_clr = [(0, 204, 102)]*151
clr_ind = 0

clr = [(0, 204, 102), (255, 0, 0),
       (0, 0, 153), (255, 102, 0)]

bigfont = pygame.font.SysFont("comicsans", 70)
fnt = pygame.font.SysFont("comicsans", 30)
fnt1 = pygame.font.SysFont("comicsans", 20)

# Sorting Algorithm: Heap Sort
def heapSort(array):
    n = len(array)

    for i in range(n//2-1, -1, -1):
        heapify(array, i, n)

    for i in range(n-1, 0, -1):
        array[i], array[0] = array[0], array[i]
        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:
        array[largest], array[root] = array[root], array[largest]
        heapify(array, largest, size)

# 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)
    heapSort(array)


# 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(200)


def binarySearch(array, key):
    left = 0
    right = len(array)-1

    while left < right:
        arr_clr[left] = clr[1]
        arr_clr[right] = clr[1]
        refill()
        refill()
        mid = left+(right-left)//2

        if array[mid] == key:
            arr_clr[left] = clr[0]
            arr_clr[right] = clr[0]
            arr_clr[mid] = clr[2]
            return 1

        elif array[mid] < key:
            arr_clr[left] = clr[0]
            left = mid+1

        else:
            arr_clr[right] = clr[0]
            right = mid-1
        refill()
    arr_clr[left] = clr[0]
    arr_clr[right] = clr[0]
    refill()
    return -1


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

    # Text should be rendered
    txt = fnt.render("SEARCH: 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("ENTER NUMBER TO SEARCH:" +
                       str(key), 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)
    if foundkey == 1:
        text4 = bigfont.render("Key Found. Press N to Reset Key", 1, (0, 0, 0))
        screen.blit(text4, (100, 300))

    elif foundkey == -1:
        text4 = bigfont.render(
            "Key Not Found. Press N to Reset Key", 1, (0, 0, 0))
        screen.blit(text4, (30, 300))


# 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:
                key = 0
                foundkey = 0
                generate_arr()
            if event.key == pygame.K_n:
                foundkey = 0
                key = 0
                for i in range(0, len(array)):
                    arr_clr[i] = clr[0]
            if event.key == pygame.K_RETURN and key != 0:
                foundkey = binarySearch(array, key)
            if event.key == pygame.K_0:
                key = key*10
            if event.key == pygame.K_1:
                key = key*10+1
            if event.key == pygame.K_2:
                key = key*10+2
            if event.key == pygame.K_3:
                key = key*10+3
            if event.key == pygame.K_4:
```

```
                key = key*10+4
            if event.key == pygame.K_5:
                key = key*10+5
            if event.key == pygame.K_6:
                key = key*10+6
            if event.key == pygame.K_7:
                key = key*10+7
            if event.key == pygame.K_8:
                key = key*10+8
            if event.key == pygame.K_9:
                key = key*10+9
    draw()
    pygame.display.update()

pygame.quit()
```

Output:



Looking to dive into the world of programming or sharpen your Python skills? Our Master Python: Complete Beginner to Advanced Course is your ultimate guide to becoming proficient in Python. This course covers everything you need to build a solid foundation from fundamental programming concepts to advanced techniques. With **hands-on projects**, real-world examples, and expert guidance, you'll gain the confidence to tackle complex **coding challenges**. Whether you're starting from scratch or aiming to enhance your skills, this course is the perfect fit. Enroll now and master Python, the language of the future!

M mano... 1

Previous Article

Sorting algorithm visualization : Insertion Sort

Next Article

Building and visualizing Sudoku Game Using Pygame

Similar Reads

Ternary Search Visualization using Pygame in Python

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

5 min read

Adding Collisions Using pygame.Rect.colliderect in Pygame

Prerequisite: Drawing shapes in Pygame, Introduction to pygame In this article, we are going to use pygame.Rect.colliderect for adding collision in a shape usin...

3 min read

Binary Search Visualization using JavaScript

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

4 min read

How to create walking character using multiple images from sprite sheet...

In this article, we will cover how to create a walking character using multiple images from a sprite sheet using Pygame. Any video or films that are made are...

5 min read

Slide Puzzle using PyGame - Python

Slide Puzzle game is a 2-dimensional game i.e the pieces can only be removed inside the grid and reconfigured by sliding them into an empty spot. The slide...

15 min read

Python - Drawing design using arrow keys in PyGame

Pygame is a cross-platform set of Python modules designed for writing video games. It includes computer graphics and sound libraries designed to be used...

3 min read

Snowfall display using Pygame in Python

Not everybody must have witnessed Snowfall personally but wait a minute, What if you can see the snowfall right on your screen by just a few lines of...

3 min read

Building Space Invaders Using PyGame - Python

In this article, we're going to build a Space Bullet shooter game using PyGame in Python. The used file can be downloaded from here. Approach: Import the...

13 min read

Brick Breaker Game In Python using Pygame

Brick Breaker is a 2D arcade video game developed in the 1990s. The game consists of a paddle/striker located at the bottom end of the screen, a ball, and...

14 min read

Snake Game in Python - Using Pygame module

Snake game is one of the most popular arcade games of all time. In this game, the main objective of the player is to catch the maximum number of fruits without...

15+ min read

Article Tags : Python Binary Search Python-projects Python-PyGame

Practice Tags : Binary Search python

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

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