Bansilal Ramnath Agarwal Charitable Trust's
Vishwakarma Institute of Information Technology

Department of
Artificial Intelligence and Data Science

Student Name: Mohammad Faiz Nishat Parvej Saiyad		
Class: TY	Division: A	Roll No:371034
Semester: V	Academic Year:2022-23	
Subject Name & Code: Design and Analysis of Algorithms		
Title of Assignment: Write a program to perform binary search on an unsorted random list of at least 5000 elements. The key element should be user input. Use the Divide & Conquer method to implement this program.		
Date of Performance:	Date of Submission:	

Aim:

Write a program to perform binary search on an unsorted random list of at least 5000 elements. The key element should be user input. Use the Divide & Conquer method to implement this program

Problem Statement:

To Implement Binary Search on unsorted Array Using Divide and Conquer method

Software Requirements:

Text Editor: VSCode, Neovim, etc

Environment: Python 3.10

Terminal Emulator

Background Information:

Divide and Conquer :-

A divide and conquer algorithm is a strategy of solving a large problem by breaking the problem into smaller sub-problems solving the sub-problems, and combining them to get the desired output.

Here are the steps involved:

1. Divide: Divide the given problem into sub-problems using recursion.
2. Conquer: Solve the smaller sub-problems recursively. If the subproblem is small enough, then solve it directly.
3. Combine: Combine the solutions of the sub-problems that are part of the recursive process to solve the actual problem.

Binary Search:-

Binary Search is a searching algorithm used in a sorted array by repeatedly dividing the search interval in half. The idea of binary search is to use the information that the array is sorted and reduce the time complexity to O(Log n).

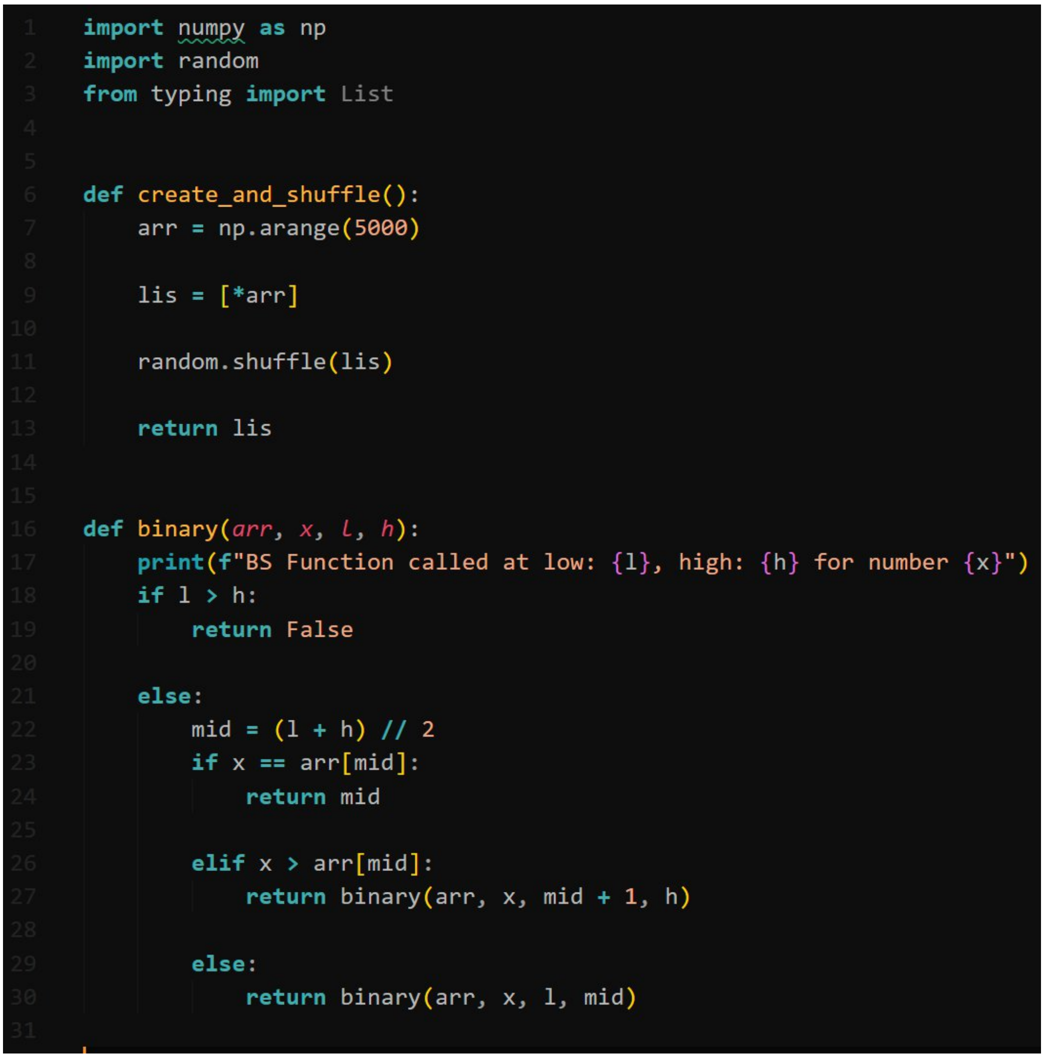
Binary Search Algorithm: The basic steps to perform Binary Search are:

- Begin with the mid element of the whole array as a search key.
- If the value of the search key is equal to the item then return an index of the search key.
- Or if the value of the search key is less than the item in the middle of the interval, narrow the interval to the lower half.
- Otherwise, narrow it to the upper half.
- Repeatedly check from the second point until the value is found or the interval is empty.

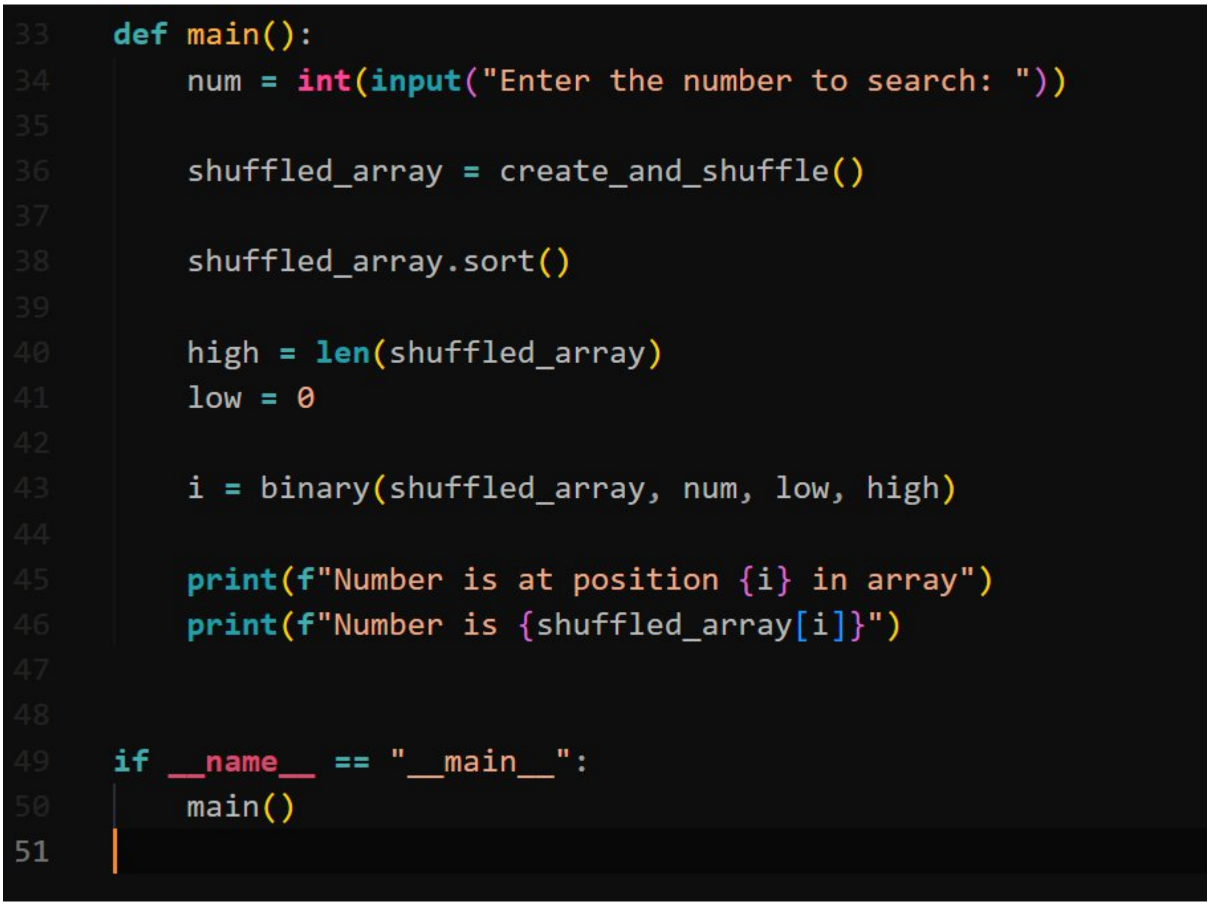
Recursive Approach for Binary Search -

```
binarySearch(arr, x, low, high)
        if low > high
        return False
    else
        mid = (low + high) / 2
        if x == arr[mid]
            return mid
        else if x > arr[mid] // x is on the right side
            return binarySearch(arr, x, mid + 1, high)
        else // x is on the right side
    return binarySearch(arr, x, low, mid - 1)
```

Code:

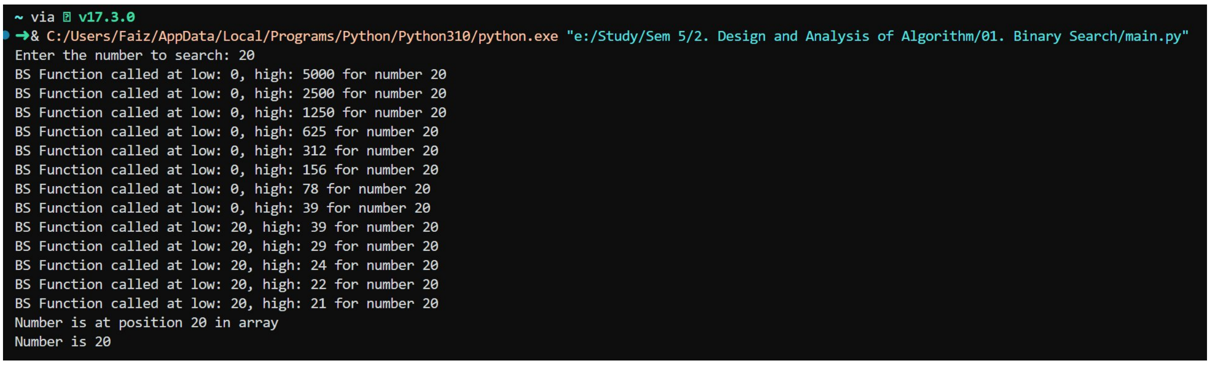


```python
import numpy as np
import random
from typing import List


def create_and_shuffle():
    arr = np.arange(5000)

    lis = [*arr]

    random.shuffle(lis)

    return lis


def binary(arr, x, l, h):
    print(f"BS Function called at low: {l}, high: {h} for number {x}")
    if l > h:
        return False

    else:
        mid = (l + h) // 2
        if x == arr[mid]:
            return mid

        elif x > arr[mid]:
            return binary(arr, x, mid + 1, h)

        else:
            return binary(arr, x, l, mid)

```



```python
def main():
    num = int(input("Enter the number to search: "))

    shuffled_array = create_and_shuffle()

    shuffled_array.sort()

    high = len(shuffled_array)
    low = 0

    i = binary(shuffled_array, num, low, high)

    print(f"Number is at position {i} in array")
    print(f"Number is {shuffled_array[i]}")


if __name__ == "__main__":
    main()

```

Output:



```
~ via □ v17.3.0
→& C:/Users/Faiz/AppData/Local/Programs/Python/Python310/python.exe "e:/Study/Sem 5/2. Design and Analysis of Algorithm/01. Binary Search/main.py"
Enter the number to search: 20
BS Function called at low: 0, high: 5000 for number 20
BS Function called at low: 0, high: 2500 for number 20
BS Function called at low: 0, high: 1250 for number 20
BS Function called at low: 0, high: 625 for number 20
BS Function called at low: 0, high: 312 for number 20
BS Function called at low: 0, high: 156 for number 20
BS Function called at low: 0, high: 78 for number 20
BS Function called at low: 0, high: 39 for number 20
BS Function called at low: 20, high: 39 for number 20
BS Function called at low: 20, high: 29 for number 20
BS Function called at low: 20, high: 24 for number 20
BS Function called at low: 20, high: 22 for number 20
BS Function called at low: 20, high: 21 for number 20
Number is at position 20 in array
Number is 20
```

Conclusion:

Implemented Binary Search using Divide and conquer Strategy on Unsorted Array