22AIE 112 Data Structures and Algorithms
Lab Sheet 1
Array

Date:13/5/2023 **RollNo:AM.EN.U4AIE22009**

1. Write a program to search for an element in a sorted array efficiently.

Code:

```
def binsearch(arr,search):
    arr.sort()
    low=0
    flag=0
    high=len(arr)-1
    while(low<=high):
        mid=(low+high)//2
        if arr[mid]==search:
            flag=1
            break
        elif arr[mid]<search:
            low=mid+1
        else:
            high=mid-1
    if flag==1:
        print(f"value found at {mid}")
    else:
        print("value not found")

arr=[]
n=int(input("enter the size of ghe array"))
print("enter the array ")
for i in range(0,n):
    e=int(input())
    arr.append(e)
search=int(input("enter the element to be searched "))
print(binsearch(arr,search))
```

Ouput:

```
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /
enter the size of ghe array4
enter the array
1
2
3
4
enter the element to be searched 3
value found at 2
None
the_architect@thearchitect:~/DSA/Labsheet1$
```

2.Write a program to search for the second occurrence of ‘6’ in an array and replace it with ‘7’.

Code:

```python
def swap(arr):
    flag=0
    i=0
    while(i<len(arr)):
        if(flag==0 and arr[i]==6):
            flag+=1
            i+=1
        elif(flag!=0 and arr[i]==6):
            arr[i]=7
            i+=1
        else:
            i+=1
    return arr
arr=[]
n=int(input())
for i in range(0,n):
    e=int(input())
    arr.append(e)
print(swap(arr))
```

Output:

```
[1, 6, 7, 7]
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /
4
1
6
6
8
[1, 6, 7, 8]
the_architect@thearchitect:~/DSA/Labsheet1$
```

3.Write a program to perform the following operations on array
a. Creation
b. Insertion (at start, at end, using index, based on value)
c. Deletion (at start, at end, using index, based on value)
d. Traversal
e. Searching an element. (based on value, based on index)

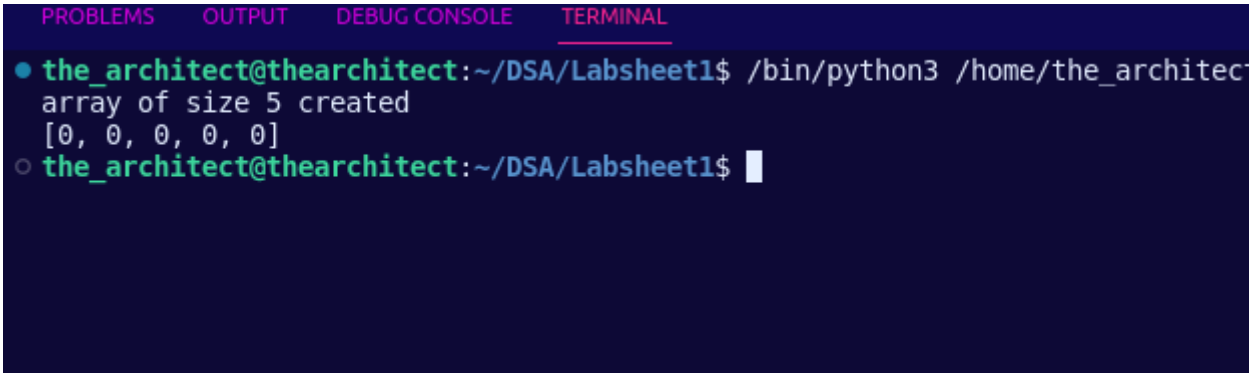
Code: a)

```
class Array:
    def __init__(self):
        self.arr=[]
    def create(self,size):
        self.arr=[0]*size
        print(f"array of size {size} created")


    def __str__(self):
        return str(self.arr)

if __name__=="__main__":
    arr=Array()
    arr.create(5)
    print(arr)
```

Output:



Code: b)

```python
class Array:
    def __init__(self):
        self.arr=[]
    def create(self,size):
        self.arr=[0]*size
        print(f"array of size {size} created")
    def insert_start(self,value):
        self.arr[0]=value
    def insert_end(self,value):
        self.arr[-1]=value
    def insert_index(self,value,index):
        self.arr[index]=value
    def insert_value(self, val):
        try:
            index = self.arr.index(val)
            print(f"{val} already exists at index {index}.")
        except ValueError:
            self.arr.append(val)
            print(f"{val} inserted based on value.")

    def __str__(self):
        return str(self.arr)

if __name__=="__main__":
    arr=Array()
    arr.create(5)
    print(arr)
    arr.insert_start(4)
    print(arr)
    arr.insert_end(4)
    print(arr)
    arr.insert_index(3,3)
    print(arr)
    arr.insert_value(5)
    print(arr)
```

Output

```
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /home/the_a
array of size 5 created
[0, 0, 0, 0, 0]
[4, 0, 0, 0, 0]
[4, 0, 0, 0, 4]
[4, 0, 0, 3, 4]
5 inserted based on value.
[4, 0, 0, 3, 4, 5]
the_architect@thearchitect:~/DSA/Labsheet1$
```

Code c)

```
class Array:
    def __init__(self):
        self.arr=[]
    def create(self,size):
        self.arr=[0]*size
        print(f"array of size {size} created")
    def delete_start(self):
        self.arr = self.arr[1:]
        print("Deleted first element.")

    def delete_end(self):
        self.arr = self.arr[:-1]
        print("Deleted last element.")

    def delete_index(self, index):
        del self.arr[index]
        print(f"Deleted element at index {index}.")

    def delete_value(self, val):
        try:
            index = self.arr.index(val)
            del self.arr[index]
            print(f"{val} deleted based on value at index
        except ValueError:
            print(f"{val} not found in array.")

    def __str__(self):
        return str(self.arr)

if __name__=="__main__":
    arr=Array()
    arr.create(5)
    print(arr)
    arr.delete_start()
    arr.delete_end()
    arr.delete_index(1)
    arr.delete_value(30)
    arr.delete_value(50)
```

Output:

```
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /home/the
array of size 5 created
[0, 0, 0, 0, 0]
Deleted first element.
Deleted last element.
Deleted element at index 1.
30 not found in array.
50 not found in array.
the_architect@thearchitect:~/DSA/Labsheet1$
```

Code d)

```
size=int(input("enter the size of the array"))
arr=[0]*size
for i in range(0,size):
    arr[i]=int(input())
print("starting array traversal")
for i in range(0,size):
    print(arr[i])
```

Output

```
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /ho
enter the size of the array4
1
2
3
4
starting array traversal
1
2
3
4
the_architect@thearchitect:~/DSA/Labsheet1$
```

Code e)

```
class Array:
    def __init__(self):
        self.arr = []
```

```
def create(self, n):
    self.arr = [0] * n
    print(f"Array of size {n} created.")

def insert_start(self, val):
    self.arr = [val] + self.arr
    print(f"{val} inserted at start.")

def insert_end(self, val):
    self.arr.append(val)
    print(f"{val} inserted at end.")

def insert_index(self, index, val):
    self.arr.insert(index, val)
    print(f"{val} inserted at index {index}.")

def insert_value(self, val):
    try:
        index = self.arr.index(val)
        print(f"{val} already exists at index {index}.")
    except ValueError:
        self.arr.append(val)
        print(f"{val} inserted based on value.")

def delete_start(self):
    self.arr = self.arr[1:]
    print("Deleted first element.")

def delete_end(self):
    self.arr = self.arr[:-1]
    print("Deleted last element.")

def delete_index(self, index):
    del self.arr[index]
    print(f"Deleted element at index {index}.")

def delete_value(self, val):
    try:
        index = self.arr.index(val)
        del self.arr[index]
        print(f"{val} deleted based on value at index {index}.")
    except ValueError:
        print(f"{val} not found in array.")

def traverse(self):
    print("Array elements:")
    for val in self.arr:
        print(val)

def search_by_value(self, val):
    try:
        index = self.arr.index(val)
```

```
            print(f"{val} found at index {index}.")
        except ValueError:
            print(f"{val} not found in array.")

    def search_by_index(self, index):
        try:
            val = self.arr[index]
            print(f"Value at index {index} is {val}.")
        except IndexError:
            print(f"Index {index} is out of range.")

if __name__ == "__main__":
    arr = Array()
    arr.create(5)
    arr.insert_start(10)
    arr.insert_end(20)
    arr.insert_index(2, 30)
    arr.insert_value(40)
    arr.insert_value(20)
    arr.delete_start()
    arr.delete_end()
    arr.delete_index(1)
    arr.delete_value(30)
    arr.delete_value(50)
    arr.traverse()
    arr.search_by_value(40)
    arr.search_by_index(2)
    arr.search_by_index(5)
```

Output:

4. Given an array with n numbers split it from a specified position, and move the first part of array and append it to the end.
Example: Given A=[12,10,5,6,22,52] and position =2
Your function should return the modified array as:[5,6,22,52,12,10]

Code :

```
arr=[]
n=int(input("enter the length "))
position=int(input("enter position to operate on "))
for i in range(0,n):
    e=int(input())
    arr.append(e)
new=arr[:position]
arr[:position]=[]
arr.extend(new)
print(arr)
```

Output:

```
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /home/the_
enter the length 6
enter position to operate on 2
12
10
5
6
22
52
[5, 6, 22, 52, 12, 10]
the_architect@thearchitect:~/DSA/Labsheet1$
```

5.Given a sorted array of nums, remove the duplicates such that each element appears only once and return the new length.
Example: Given nums = [0,0,1,1,1,2,2,3,3,4]
Your function should return length = 5, with the first five elements being modified as 0, 1, 2, 3, and 4 respectively.

Code:

```
arr=[]
n=int(input())
for i in range(0,n):
    e=int(input())
    arr.append(e)
print("lenght of the editted array is",end=" ")
print(len(set(arr)))
print("the array is ", list(set(arr)))
```

Output:

```
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /home/the_
4
1
2
2
4
lenght of the editted array is 3
the array is  [1, 2, 4]
the_architect@thearchitect:~/DSA/Labsheet1$
```

6.Given an array of integers, return indices of the two numbers such that they add up to a specific target. You may assume that each input would have exactly one solution, and you may not use the same element twice.
Example: Given nums = [2, 7, 11, 15], target = 9,
Because nums[0] + nums[1] = 2 + 7 = 9, then return [0, 1]

Code:

```
def sumfind(nums, target):
    for i in range(0,len(nums)):
        for j in range(i+1,len(nums)):
            if nums[i]+nums[j]==target:
                return[i,j]
    return []

arr=[]
n=int(input("enter the length of the array "))
target=int(input("enter the target value"))
for i in range(0,n):
    e=int(input())
    arr.append(e)
print(sumfind(arr,target))
```

Output:

```
the array is  [1, 2, 4]
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /home/the_a
enter the length of the array 5
enter the target value3
1
2
4
3
5
[0, 1]
the_architect@thearchitect:~/DSA/Labsheet1$
```

7.Given an array nums and a value val, remove all instances of that value in the array and return the new length. The order of elements can be changed.
Example: Given nums = [0,1,2,2,3,0,4,2], val = 2,
Your function should return length = 5, with the first five elements of nums Containing 0, 1, 3, 0, and 4.

Code:

```
def popper(arr, value):
    for i in range(len(arr)-1,-1,-1):
        if arr[i]==value:
            arr.pop(i)
    return(len(arr))

arr=[]
n = int(input("enter the size of array "))
value=int(input("enter the value to be removed "))
for i in range(0,n):
    e=int(input())
    arr.append(e)
print("new length is ",popper(arr,value))
```

Output:

```
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /home/the_archite
enter the size of array 5
enter the value to be removed 2
1
2
3
4
4
new length is  4
the_architect@thearchitect:~/DSA/Labsheet1$
```

8.Given an array of n elements to find if an integer x appears more than n/2 times in a sorted array of n integers.
Example: Given A=[0,1,2,4,4,4,4,4]
Your function should return 4 appears 5 times

Code:

```
def occ_count(arr, x):
    count = 0
    for e in arr:
        if e == x:
            count += 1
    return count

def major(arr, x):
    count = occ_count(arr, x)
    if count > len(arr) // 2:
        return f"{x} appears {count} times in the array."
    else:
        return f"{x} does not appear more than n/2 times in the array."

arr=[]
n = int(input("enter the size of array "))
value=int(input("enter the value to be searched "))
for i in range(0,n):
    e=int(input())
    arr.append(e)
print(major(arr,value))
```

Output

```
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /home/the_architec
enter the size of array 5
enter the value to be searched 4
1
2
4
4
4
4 appears 3 times in the array.
the_architect@thearchitect:~/DSA/Labsheet1$
```

9. Write a program to merge elements of two sorted arrays A and B of size p and q, by maintaining the sorted order i.e. fill A with first p smallest elements and fill B with remaining elements.
Example: Input :
intA[] = { 1, 5, 6, 7, 8, 10 },int B[] = { 2, 4, 9 }
Output: Sorted Arrays:
A: [1, 2, 4, 5, 6, 7], B: [8, 9, 10]

Code:

```
input_string = input("Enter the array separated by commas: ")
input_list = input_string.split(",")
arrA = [int(num) for num in input_list]

input_string = input("Enter the next array separated by commas: ")
input_list = input_string.split(",")
arrB = [int(num) for num in input_list]

arrA.sort()
arrB.sort()

p = len(arrA)
q = len(arrB)

arrA += arrB  # append the flattened list
arrA.sort()

print("Sorted Arrays:")
print("A:", arrA[:p])
print("B:", arrA[p:])
```

Output:

```
4
4 appears 3 times in the array.
the_architect@thearchitect:~/DSA/Labsheet1$ /bin/python3 /home/the_architec
Enter the array separated by commas: 1, 5, 6, 7, 8, 10
Enter the next array separated by commas: 2, 4, 9
Sorted Arrays:
A: [1, 2, 4, 5, 6, 7]
B: [8, 9, 10]
the_architect@thearchitect:~/DSA/Labsheet1$
```