1: Perform Crud operation in an array.

**Solution**

CURD = Create, Update, Read, Delete.

**Algorithm for CRUD Operations on an Array**

**1️ Create (Insert an element into the array)**

1. If the array is full (in case of a fixed-size array), return an error.
2. Add the new element at the next available position.
3. Increase the size of the array.

**2️ Read (Access elements in the array)**

1. Loop through the array or access an element directly using its index.
2. If the index is out of bounds, return an error.
3. Display the element(s).

**3️ Update (Modify an element in the array)**

1. Check if the given index is within the valid range.
2. Replace the element at the specified index with a new value.

**4️ Delete (Remove an element from the array)**

1. Find the index of the element to be deleted.
2. Shift all elements after the deleted element one position to the left.
3. Decrease the size of the array.

**Example of CRUD Operations of array**

**# Create**

arr = [10, 20, 30, 40]

print("Original Array:", arr)

**# Read**

print("Element at index 2:", arr[2])

**# Update**

arr[1] = 25

print("Updated Array:", arr)

**# Delete**

arr.remove(30)

print("Array after deletion:", arr)

**CURD in one Code**

arr=[]

def insert(arr,val,maxsize):

if(len(arr)<maxsize):

arr.append(val)

return arr

else:

print("Not inserted, max size reached")

return arr

def read(arr):

print("")

print(\*arr,sep=" ")

def update(arr,val,index):

if(index<len(arr) and index>=0):

arr[index]=val

return arr

else:

print("Index out of range")

return arr

def delete(arr,val):

if(val in arr):

arr.remove(val)

return arr

else:

print("Element not found")

return arr

exit=False

maxsize=int(input("Enter the max size of the array: "))

while(not exit):

print("")

print(1,"Insert")

print(2,"Read")

print(3,"Update")

print(4,"Delete")

print(5,"Exit")

inp=int(input("Enter the operation to be performed: "))

if(inp==5):

exit=True

elif(inp==1):

val=int(input("Enter the value to be inserted: "))

arr=insert(arr,val,maxsize)

print("inserted")

elif(inp==2):

read(arr)

elif(inp==3):

val=int(input("Enter the new value: "))

index=int(input("Enter the index to be updated: "))

arr=update(arr,val,index)

print("Updated")

elif(inp==4):

val=int(input("Enter the value to be deleted: "))

arr=delete(arr,val)

print("Deleted")

else:

print("Invalid input")

2: Take user input of numbers to perform linear search in an array or list

**SOLUTION**

**Algorithm for Linear Search**

**1️ Take Input from the User:**

* Ask the user to enter a list of numbers (separated by spaces).
* Ask the user for the number they want to search for.

**2️ Start Searching:**

* Begin from the first element and compare it with the target number.
* If they match, print the position and stop searching.
* If not, move to the next element and repeat.

**3️ Check the Result:**

* If the number is found, display its position.
* If we reach the end and don’t find the number, print "Not found."

**Pseudocode**

1. Take user input for numbers and store them in a list.
2. Ask the user for the target number they want to find.
3. Loop through each number in the list:
   1. If the number matches the target, print its position and exit.
4. If the loop ends without finding the number, print "Number not found.”

**Code**

def linear\_search(arr,val):

for i in range(len(arr)):

if arr[i] == val:

return "found at position {}".format(i+1)

return "not found"

arr=list(map(int,input("Enter numbers separated by spaces:").strip("").split(" ")))

val=int(input("Enter the value to search for: "))

print("Number {}".format(linear\_search(arr, val)))

**Example Run 1**

Enter numbers separated by spaces: 10 20 30 40 50

Enter the number to search for: 30

Number found at position 3

**Example Run 2**

Enter numbers separated by spaces: 5 15 25 35

Enter the number to search for: 100

Number not found.