Adamson University

College of Engineering

Computer Engineering Department

Data Structure and Algorithm

Laboratory Activity No. 4

Arrays

Submitted by:

Manigbas, Jeus Miguel T. <Leader>

Barredo, Alwin P.

Bela, Lorenzo Miguel D.

Callorina, Rovic Victor A.

Guzon, Kean Loiz I.

Instructor:

Engr. Maria Rizette H. Sayo

November, 15, 2023

I. Objectives

Introduction

Array, in general, refers to an orderly arrangement of data elements. Array is a type of data structure that stores data elements in adjacent locations. Array is considered as linear data structure that stores elements of same data types. Hence, it is also called as a linear homogenous data structure.

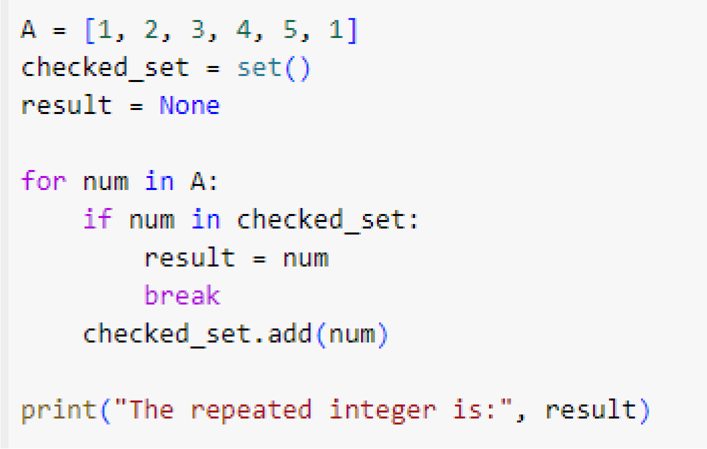
This laboratory activity aims to implement the principles and techniques in:

- Writing algorithms using Array data structure
- Writing a python program that can implement Caesar cipher using Array data structure

II. Methods

- Let A be an array of size n ≥ 2 containing integers from 1 to n−1, inclusive, with exactly one repeated. Describe a fast algorithm for finding the integer in A that is repeated.
- Write a program that can perform the Caesar cipher for English messages that include both upper- and lowercase characters.

III. Results



```
A = [1, 2, 3, 4, 5, 1]
checked_set = set()
result = None

for num in A:
    if num in checked_set:
        result = num
        break
    checked_set.add(num)

print("The repeated integer is:", result)
```

Figure 1.1: Python program

- A = [1, 2, 3, 4, 5, 1]

- In this part of the program, The input array **‘A’** is given, which is an array of integers with the requirement that there is exactly one repeated integer.

- checked_set = set()
  result = None

- The line **‘checked_set = set()’** is a set used to keep track of the numbers that have been encountered so far.

- While in the line **‘result = None’** will be used to store the repeated integer once it is found.

- for num in A:

- In this part, this is utilized as a loop iterating through each element **'num'** in the array **'A'**.

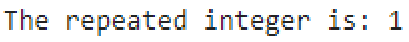
- if num in checked_set:
  result = num
  Break

- In this part, if the current number is already present in the set, which is the **'checked_set'**, it means we have found a repeated integer.

- Break out of the loop since we have already found the repeated integer.

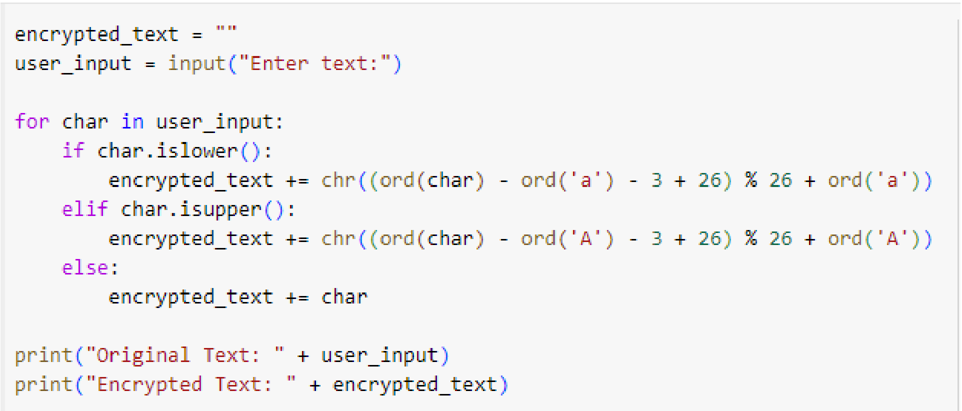
- print("The repeated integer is:", result)

- In the last part, this will print the repeated integer. If there are no integers found, the printed result will be 'None'.



```
The repeated integer is: 1
```

Figure 1.2: Output of the program



```python
encrypted_text = ""
user_input = input("Enter text:")

for char in user_input:
    if char.islower():
        encrypted_text += chr((ord(char) - ord('a') - 3 + 26) % 26 + ord('a'))
    elif char.isupper():
        encrypted_text += chr((ord(char) - ord('A') - 3 + 26) % 26 + ord('A'))
    else:
        encrypted_text += char

print("Original Text: " + user_input)
print("Encrypted Text: " + encrypted_text)
```

Figure 2.1: Python program that can perform Ceasar cypher

- The line user_input = input("Enter text:") prompts the user to enter a text and stores the input in the variable **‘user_input’**.

- encrypted_text = ""
  for char in user_input:

- This initializes an empty string **‘encrypted_text’** to store the encrypted result. The loop iterates through each character in the user's input.

- if char.islower():
  encrypted_text += chr((ord(char) - ord('a') - 3 + 26) % 26 + ord('a'))

- In this part of the program, if the current character is a lowercase letter, it gets encrypted using the formula **'(ord(char) - ord('a') - 3 + 26) % 26 + ord('a')'**. This shifts the letter three positions backward in the alphabet, taking into account the wrap-around.

- If the current character is an uppercase letter, a similar encryption formula is applied to shift it three positions backward in the uppercase alphabet.

- else:

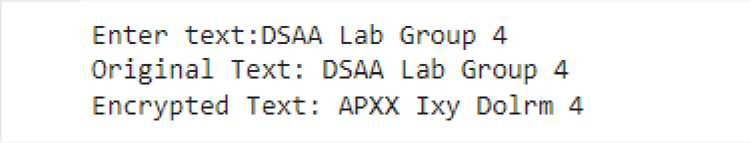
  encrypted_text += char

- In this part, if the current character is not a letter, it gets appended to the **'encrypted_text'** unchanged.

- print("Original Text: " + user_input)

  print("Encrypted Text: " + encrypted_text)

- In the end, the initial input and the encrypted outcome are displayed or printed.



```
Enter text:DSAA Lab Group 4
Original Text: DSAA Lab Group 4
Encrypted Text: APXX Ixy Dolrm 4
```

Figure 2.2: Output of the python program

IV. Conclusion

We have created a solution to identify the duplicate integer in an array during the initial phase of the program. Using a set called "checked_set" allows for efficient tracking of encountered numbers, while the variable "result" is used to store any repeated integers that are found. As the loop goes through each item in the array, it checks if a number is already in the set, signaling the presence of a duplicated integer and causing the loop to end. At last, the program will output the duplicated integer, or 'None' if none is discovered.

We then proceeded to develop a Python program for Caesar cipher encryption in the following section. The user is asked to enter text, while an empty string (encrypted_text) is initialized to hold the encrypted output, then it goes through each character in the user's input. To encrypt the text, simply shift each letter three positions backward in the alphabet, taking into account both uppercase and lowercase letters and wrapping around if necessary. Non-alphabetic characters will not be altered. At the end of the program, it prints out both the original and encrypted texts.

In conclusion, the first program aims to locate a duplicate integer within an array, while the second program introduces a simple Caesar cipher encryption. All of the programs showcase the skillful implementation of loops, conditionals, and string manipulation to accomplish their unique objectives.

References

[1] GeeksforGeeks. "Python Arrays." Sept. 06, 2023 [Online]. Available: https://www.geeksforgeeks.org/python-arrays/ [Accessed: November 15, 2023].

[2] Scaler. "Caesar Cipher in Python." May 04 2023 [Online]. Available:

https://www.scaler.com/topics/caesar-cipher-python/ [Accessed: November 15, 2023].