K.R.Nikhil Karthik
VU22CSEN0100010

ASSIGNMENT-1
INTERMEDIATE CODING

1)Remove an element: Given an integer array arr and an integer key, write a program in C to return the number of values not equal to key.

Code:

```
def remove_element(arr, key):
    count = 0
   for i in range(len(arr)):
        if arr[i] != key:
            arr[count] = arr[i]
            count += 1
    return count

def main():
    # Take input for the list of integers
    arr = list(map(int, input("Enter the array elements separated
by spaces: ").split()))

    key = int(input("Enter the key to remove: "))

    new_length = remove_element(arr, key)

    print(f"Number of elements not equal to {key}:
{new_length}")
    print("Updated array:", end=" ")

    for i in range(new_length):
        print(arr[i], end=" ")

    print()
```

```
if __name__ == "__main__":
    main()
```

```
Enter the array elements separated by spaces: 2 3 4  5 6
Enter the key to remove: 5
Number of elements not equal to 5: 4
Updated array: 2 3 4 6
```

2) Remove duplicates from a given array: Given a sorted integer array arr, write a program in C to return the array after removing the duplicates, with all the other elements in place.

Code:

```
def remove_duplicates(arr):
    return list(dict.fromkeys(arr))

arr = list(map(int, input("Enter the elements of the array separated by spaces: ").split()))

result = remove_duplicates(arr)

print("Array after removing duplicates:", result)
```

```
Enter the elements of the array separated by spaces: 1 2 2 3 3 4 5 6 7
Array after removing duplicates: [1, 2, 3, 4, 5, 6, 7]

=== Code Execution Successful ===
```

3) Best time to sell a commodity: Given an array of positive integers that represent the price of a commodity on a given day, return two integers which represent the best day to buy and later, sell the commodity.

Code:

```
def best_time_to_sell(prices):
    if not prices:
        return -1, -1
    min_price = prices[0]
    min_day = 0
```

```
    max_profit = 0
    buy_day, sell_day = 0, 0
    for i in range(1, len(prices)):
        if prices[i] - min_price > max_profit:
            max_profit = prices[i] - min_price
            buy_day =
min_day
            sell_day = i
        if prices[i] <
min_price:
            min_price =
prices[i]
            min_day = i
    return buy_day, sell_day
prices = [7, 1, 5, 3, 6, 4]
buy, sell = best_time_to_sell(prices)
print(f"Buy on day {buy} and sell on day {sell}")
```

```
Buy on day 1 and sell on day 4

=== Code Execution Successful ===
```

4) Merging the strings: Given two strings str1 and str2 in two files file1 and file2, write a program to create a third string str3 with one character from each array alternatively. If one string is longer than the other, append the extra characters to str3. Write str3 into a different text file called file3.

Code:

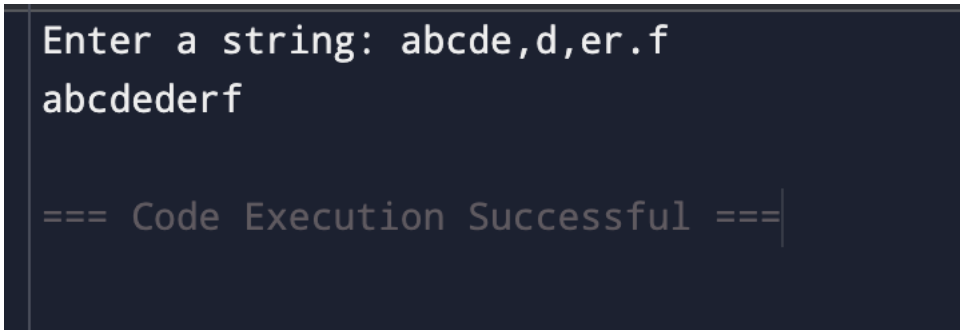
```
def merge_strings(file1, file2, output_file):
    with open(file1, 'r') as f1, open(file2, 'r') as f2:
        str1 = f1.read().strip()
        str2 = f2.read().strip()

    str3 = []
    len1, len2 = len(str1), len(str2)
```

```
    for i in range(max(len1, len2)):
        if i < len1:
            str3.append(str1[i])
        if i < len2:
            str3.append(str2[i])

    with open(output_file, 'w') as f3:
        f3.write(''.join(str3))

merge_strings('file1.txt', 'file2.txt', 'file3.txt')
print("Merged string written to file3.txt")
```

5) String reversal: Given an string with words separated by empty spaces, commas or fullstops, write a program to return the string after removing the commas and fullstops and reversing the words in the string.

Code:

```
import re

def reverse_string(s):

    clean_string = re.sub(r'[,.]', '', s)
    words = clean_string.split()
    reverse_string = ' '.join(reversed(words))

    return reverse_string

input_string = input("Enter a string: ")

result = reverse_string(input_string)
print(result)
```



K.R.Nikhil Karthik
VU22CSEN0100010

6) Move all zeros to the end: Given an integer array arr, return the array with all the zeros moved to the end of the array. The relative ordering of the non-zero lements should not change.

Code:

```
def remove_zeros(arr):
    # Count the number of zeros in the array
    zero_count = arr.count(0)

    # Create a list of non-zero elements
    k = [x for x in arr if x != 0]

    # Add the zeros at the end
    k.extend([0] * zero_count)

    return k

# Input array
arr = list(map(int, input("Enter array elements separated by space:
").split()))

# Calling the function
result = remove_zeros(arr)

# Printing the result
print(result)
```

```
Enter array elements separated by space: 1 0 3 2 0 3 30 4 0 4 30 0 20
[1, 3, 2, 3, 30, 4, 4, 30, 20, 0, 0, 0, 0]

=== Code Execution Successful ===
```

7) Is substring: Given two strings str1 and str2, write a program to return true if str2 is a substring of str1 and false otherwise.

Code:

```
def is_substring(main_str, sub_str):
    if sub_str in main_str:
        return sub_str
    else:
        return False

main_str = input("enter a string")

sub_str =  input("enter a sub_string")
```

```
result = is_substring(main_str, sub_str)
print(result)
```

```
Enter the main string: nikhil
Enter the substring to check: nik
nik
```