

# **ASSIGNMENT**

**Objective:** Implementation and Analysis of Radix sort using python

## **Code:**

```
def radix_sort(alist, base=10):  
    if alist == []:  
        return  
  
    def key_factory(digit, base):  
        def key(alist, index):  
            return ((alist[index]//(base**digit)) % base)  
        return key  
  
    largest = max(alist)  
    exp = 0  
    while base**exp <= largest:  
        alist = counting_sort(alist, base - 1, key_factory(exp, base))  
        exp = exp + 1  
  
    return alist  
  
def counting_sort(alist, largest, key):  
    c = [0]*(largest + 1)  
    for i in range(len(alist)):  
        c[key(alist, i)] = c[key(alist, i)] + 1  
  
    c[0] = c[0] - 1 # to decrement each element for zero-based indexing  
  
    for i in range(1, largest + 1):  
        c[i] = c[i] + c[i - 1]  
  
    result = [None]*len(alist)  
  
    for i in range(len(alist) - 1, -1, -1):
```

```
result[c[key(alist, i)]] = alist[i]
```

```
c[key(alist, i)] = c[key(alist, i)] - 1
```

```
return result
```

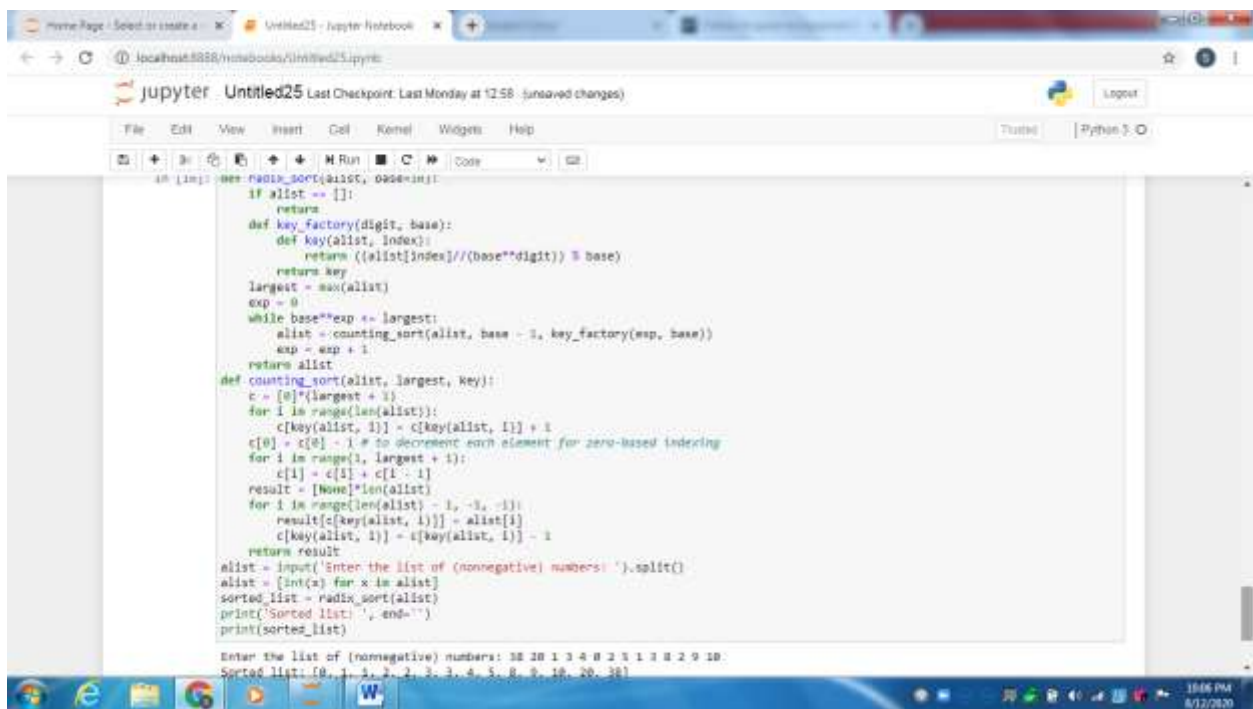
```
alist = input('Enter the list of (nonnegative) numbers: ').split()
```

```
alist = [int(x) for x in alist]
```

```
sorted_list = radix_sort(alist)
```

```
print('Sorted list: ', end='')
```

```
print(sorted_list)
```



The screenshot shows a Jupyter Notebook window titled 'Untitled25'. The interface includes a menu bar (File, Edit, View, Insert, Cell, Kernel, Widgets, Help) and a toolbar with icons for running, saving, and other actions. The code cell contains a Python implementation of radix sort. The code defines a `radix_sort` function that uses a `key_factory` to generate keys for each digit. It then uses a `counting_sort` function to sort the list by each digit. The `counting_sort` function uses a counting array `c` to count the frequency of each key and then places the elements into the sorted list. The main code block takes user input, converts it to a list of integers, and prints the sorted list. The output shows the sorted list: `[0, 1, 1, 2, 2, 2, 3, 3, 3, 4, 5, 8, 9, 10, 20, 38]`.

```
def radix_sort(alist, base=10):
    if alist == []:
        return
    def key_factory(digit, base):
        def key(alist, index):
            return ((alist[index] // (base**digit)) % base)
        return key
    largest = max(alist)
    exp = 0
    while base**exp <= largest:
        alist = counting_sort(alist, base=base, key_factory=key_factory(exp, base))
        exp = exp + 1
    return alist
def counting_sort(alist, largest, key):
    c = [0]*(largest + 1)
    for i in range(len(alist)):
        c[key(alist, i)] = c[key(alist, i)] + 1
    c[0] = c[0] - 1 # to decrement each element for zero-based indexing
    for i in range(1, largest + 1):
        c[i] = c[i] + c[i-1]
    result = [None]*len(alist)
    for i in range(len(alist) - 1, -1, -1):
        result[c[key(alist, i)] - 1] = alist[i]
        c[key(alist, i)] = c[key(alist, i)] - 1
    return result
alist = input('Enter the list of (nonnegative) numbers: ').split()
alist = [int(x) for x in alist]
sorted_list = radix_sort(alist)
print('Sorted list: ', end='')
print(sorted_list)
```

Enter the list of (nonnegative) numbers: 0 1 1 2 2 2 3 3 3 4 5 8 9 10 20 38  
Sorted list: [0, 1, 1, 2, 2, 2, 3, 3, 3, 4, 5, 8, 9, 10, 20, 38]