TY. B. Tech.

Design & Analysis of Algorithm

Assignment No: 2

Date of Submission: 15/04/2023

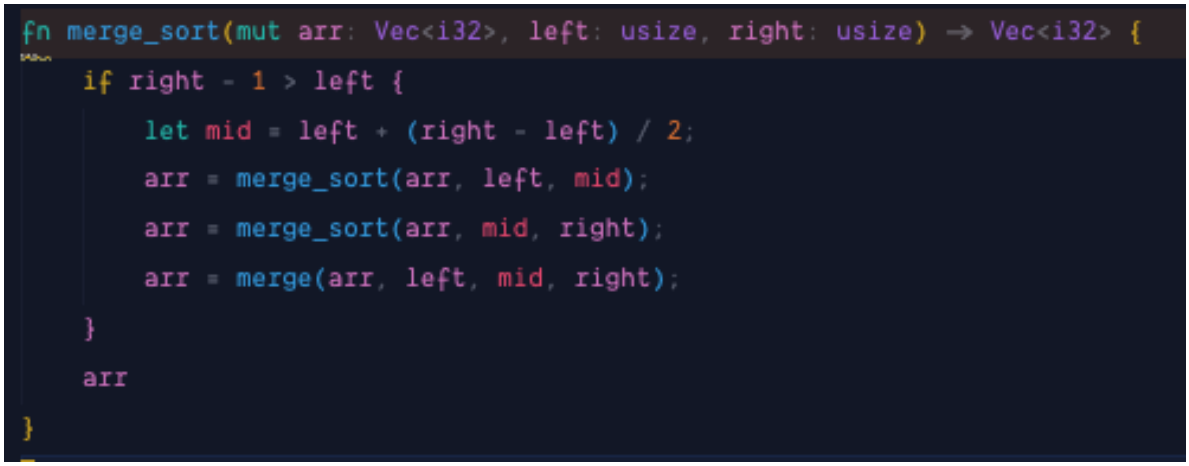
Roll. No.	Gr. No.	Div	Name
12	12011336	C	Niraj Kirit Patil

Academic Year: 2021-22 **Semester: II**

Assignment No: 2

Merge Sort

Code:



```rust
fn merge_sort(mut arr: Vec<i32>, left: usize, right: usize) -> Vec<i32> {
    if right - 1 > left {
        let mid = left + (right - left) / 2;
        arr = merge_sort(arr, left, mid);
        arr = merge_sort(arr, mid, right);
        arr = merge(arr, left, mid, right);
    }
    arr
}
```

```rust
fn merge(mut arr: Vec<i32>, left: usize, mid: usize, right: usize) -> Vec<i32> {
    let n1 = mid - left;
    let n2 = right - mid;
    let mut L1 = arr.clone();     variable does not need to be mutable⏎`#[warn(unused_mut)]` on by default
    let mut R1 = arr.clone();     variable does not need to be mutable
    let L = &L1[left..mid];     variable `L` should have a snake case name
    let R = &R1[mid..right];     variable `R` should have a snake case name
    /* Merge the temp arrays back into arr[l..r]*/
    let mut i = 0; // Initial index of first subarray
    let mut j = 0; // Initial index of second subarray
    let mut k = left; // Initial index of merged subarray
    while i < n1 && j < n2 {
        if L[i] < R[j] {
            arr[k] = L[i];
            i = i + 1;
        } else {
            arr[k] = R[j];
            j = j + 1;
        }
        k = k + 1;
    }
    while i < n1 {
        arr[k] = L[i];
        i = i + 1;
        k = k + 1;
    }
    /* Copy the remaining elements of R[], if there
    are any */
    while j < n2 {
        arr[k] = R[j];
        j = j + 1;
        k = k + 1;
    }
    arr
} fn merge
```

```rust
▶ Run | Debug
fn main() {
    let mut arr: Vec<i32> = vec![64, 34, 25, 8, 22, 11, 9];
    arr = merge_sort(arr.clone(), 0, arr.len());
    println!("Sorted array is {:?}", arr);
}
```

Output:

```
Sorted array is [8, 9, 11, 22, 25, 34, 64]
```

(This page marks the end of the assignment)