Tugas Job Sheet 9 Selection and Merge Sort_Praktikum Struktur data
Muhamad Akbar Fauzan_23343075

A. Contoh aplikasi dalam Selection Sort

```
#include <stdio.h>

void selectionSort(int arr[], int n) {
   int i, j, minIndex, temp;
   for (i = 0; i < n-1; i++) {
      minIndex = i;
      for (j = i+1; j < n; j++) {
         if (arr[j] < arr[minIndex]) {
            minIndex = j;
         }
      }
      temp = arr[i];
      arr[i] = arr[minIndex];
      arr[minIndex] = temp;
   }
}

int main() {
   int arr[] = {64, 25, 12, 22, 11};
   int n = sizeof(arr)/sizeof(arr[0]);
   printf("Array sebelum sorting: \n");
   for (int i=0; i < n; i++)
      printf("%d ", arr[i]);
   printf("\n");
   selectionSort(arr, n);
   printf("Array setelah sorting: \n");
   for (int i=0; i < n; i++)
      printf("%d ", arr[i]);
   printf("\n");
   return 0;
}
```

Penjelasan :
Selection Sort: Metode ini bekerja dengan memilih elemen terkecil dari array yang belum diurutkan dan menukarnya dengan elemen di awal array yang belum diurutkan. Ini dilakukan

secara berulang hingga seluruh array terurut. Kode di atas melakukan iterasi melalui array, mencari elemen terkecil, dan menukarnya dengan elemen pertama. Proses ini berlanjut untuk setiap elemen di dalam array.

B. Contoh aplikasi dalam Merge Sort

```
#include <stdio.h>

void merge(int arr[], int l, int m, int r) {
   int i, j, k;
   int n1 = m - l + 1;
   int n2 = r - m;

   int L[n1], R[n2];

   for (i = 0; i < n1; i++)
      L[i] = arr[l + i];
   for (j = 0; j < n2; j++)
      R[j] = arr[m + 1 + j];

   i = 0;
   j = 0;
   k = l;
   while (i < n1 && j < n2) {
      if (L[i] <= R[j]) {
         arr[k] = L[i];
         i++;
      } else {
         arr[k] = R[j];
         j++;
      }
      k++;
   }

   while (i < n1) {
      arr[k] = L[i];
      i++;
      k++;
   }

   while (j < n2) {
      arr[k] = R[j];
      j++;
      k++;
   }
}
```

```
void mergeSort(int arr[], int l, int r) {
   if (l < r) {
      int m = l + (r - l) / 2;
      mergeSort(arr, l, m);
      mergeSort(arr, m + 1, r);
      merge(arr, l, m, r);
   }
}

int main() {
   int arr[] = {12, 11, 13, 5, 6, 7};
   int arr_size = sizeof(arr) / sizeof(arr[0]);

   printf("Array sebelum sorting: \n");
   for (int i = 0; i < arr_size; i++)
      printf("%d ", arr[i]);
   printf("\n");

   mergeSort(arr, 0, arr_size - 1);

   printf("Array setelah sorting: \n");
   for (int i = 0; i < arr_size; i++)
      printf("%d ", arr[i]);
   printf("\n");
   return 0;
}
```

Penjelasan :

Merge Sort: Metode ini bekerja dengan membagi array menjadi dua bagian, mengurutkan masing-masing bagian tersebut secara rekursif, dan kemudian menggabungkan kembali dua bagian tersebut menjadi satu. Proses ini berlanjut secara rekursif hingga array terurut secara keseluruhan. Kode di atas menggunakan pendekatan rekursif untuk membagi array menjadi dua bagian, kemudian menggabungkan kembali dua bagian tersebut secara berurutan hingga array terurut secara keseluruhan.