Nama : ANAK AGUNG BRAMASTA JAYA

NIM : 1203230007

Kelas : IF-03-02

TUGAS ALPRO & STRUKTUR DATA

1. SOURCE CODE

```
#include <stdio.h>
#include <stdlib.h>

typedef struct Node *NodePointer;

struct Node {
   int value;
   NodePointer next;
   NodePointer prev;
} *head = NULL, *tail = NULL;

NodePointer createNode(int val) {
   NodePointer temp = (NodePointer)malloc(sizeof(struct Node));
   temp->value = val;
   temp->next = NULL;
   temp->prev = NULL;
   return temp;
}
//pada program diatas nodepointer akan mengalokasikan memori sebagai indeks node, menginisialisasikan nilai node menjadi NULL
void insert_last(int val) {
   NodePointer temp = createNode(val);
   if (head == NULL) {
      head = tail = temp;
      head->next = head;
      head->prev = head;
   } else {
      temp->next = head;
      temp->prev = tail;
      tail->next = temp;
      head->prev = temp;
      tail = temp;
   }
}
//pada fungsi diatas program melakukan pemasukan nilai yang telah kita inputkan ke dalam node tertentu, node head, tail dan node yang telah kita buat sebelumnya, node head->next akan menuju pada temp next apabila head tidak NULL, tail->next akan mengambil nilai temp dan seterusnya
```

```
void swap_nodes(NodePointer a, NodePointer b) {
   if (a == b) return;

   NodePointer aPrev = a->prev;
   NodePointer aNext = a->next;
   NodePointer bPrev = b->prev;
   NodePointer bNext = b->next;

   if (a->next == b) {
      a->next = bNext;
      a->prev = b;
      b->next = a;
      b->prev = aPrev;

      if (aPrev != NULL) aPrev->next = b;
      if (bNext != NULL) bNext->prev = a;
   } else if (b->next == a) {
      b->next = aNext;
      b->prev = a;
      a->next = b;
      a->prev = bPrev;

      if (bPrev != NULL) bPrev->next = a;
      if (aNext != NULL) aNext->prev = b;
   } else {

      a->next = bNext;
      a->prev = bPrev;
      b->next = aNext;
      b->prev = aPrev;

      if (aNext != NULL) aNext->prev = b;
      if (aPrev != NULL) aPrev->next = b;
      if (bNext != NULL) bNext->prev = a;
      if (bPrev != NULL) bPrev->next = a;
   }

   if (head == a) {
      head = b;
   } else if (head == b) {
      head = a;
   }

   if (tail == a) {
      tail = b;
   } else if (tail == b) {
      tail = a;
   }
```

```
}
```

Fungsi swaps_node menukar posisi node, dengan memerika kedua nya fungsi juga menyimpan prev dan next dari kedua node tersebut. Fungsi juga memastikan head dan tail akan di perbarui jika salah satu node yang ditukar adalah head atau tail.

```
void sort_ascending() {
  if (head == NULL) return;

  int swapped;
  NodePointer ptr1;
  NodePointer lptr = NULL;

  do {
    swapped = 0;
    ptr1 = head;

    do {
      if (ptr1->next != head && ptr1->value > ptr1->next->value) {
        swap_nodes(ptr1, ptr1->next);
        swapped = 1;
      }
      ptr1 = ptr1->next;
    } while (ptr1->next != head);

    lptr = ptr1;
  } while (swapped);
}
```

//Fungsi diatas sort_ascending() melakukan pengurutan node dengan memanfaatkan address pada node yang ada, metode pengurutan yang dilakukan adalah bubble sort

```
void print_list() {
  if (head == NULL) return;

  NodePointer temp = head;
  do {
    printf("Address: %p, Data: %d\n", (void*)temp, temp->value);
    temp = temp->next;
  } while (temp != head);
  printf("\n");
}

//fungsi print_list() berfungsi untuk menampilkan semua node yang telah diurutkan
int main() {
  int num_elements, i, jumlah;

  scanf("%d", &num_elements);
```

```
    if (num_elements < 1 || num_elements > 10) {
        printf("Jumlah data harus antara 1 dan 10.\n");
        return 1;
    }

    for (i = 0; i < num_elements; i++) {
        printf("", i + 1);
        scanf("%d", &jumlah);
        insert_last(jumlah);
    }

    printf("\n");
    print_list();

    sort_ascending();

    print_list();

    return 0;
}
```

2. Output

```
C:\Notepad++>gcc new1.c

C:\Notepad++>a
5
5
3
8
1
6

Address: 00000000007613F0, Data: 5
Address: 0000000000761410, Data: 3
Address: 0000000000761430, Data: 8
Address: 0000000000761450, Data: 1
Address: 0000000000761470, Data: 6

Address: 0000000000761450, Data: 1
Address: 0000000000761410, Data: 3
Address: 00000000007613F0, Data: 5
Address: 0000000000761470, Data: 6
Address: 0000000000761430, Data: 8
```