DATA STRUCTURES AND ALGORITHMS LAB – 6

Student Details

- Name: Pratyush Deo Singh
- Registration Number: 25BCE5101
- Course / Branch: B. Tech. CSE (Core)
- Semester: 2nd
- Subject: Data Structures and Algorithms
- Faculty Name: Dr Malini A

Program 1: Menu Driven Singly Linked List

Program Code

1_singly_linked_list_menu.c

```
1   #include <stdio.h>
2   #include <stdlib.h>
3
4   struct node {
5   int data;
6   struct node *next;
7   };
8
9   struct node *head = NULL;
10
11  void insert_begin() {
12  int x;
13  struct node *n = malloc(sizeof(struct node));
14  scanf("%d", &x);
15  n->data = x;
16  n->next = head;
17  head = n;
18  }
19
20  void insert_end() {
21  int x;
22  struct node *n = malloc(sizeof(struct node));
23  scanf("%d", &x);
24  n->data = x;
25  n->next = NULL;
26
```

```
27  if (head == NULL) {
28  head = n;
29  return;
30  }
31
32  struct node *t = head;
33  while (t->next != NULL)
34  t = t->next;
35  t->next = n;
36  }
37
38  void insert_pos() {
39  int x, pos, i = 1;
40  scanf("%d %d", &x, &pos);
41
42  if (pos == 1) {
43  insert_begin();
44  return;
45  }
46
47  struct node *n = malloc(sizeof(struct node));
48  n->data = x;
49
50  struct node *t = head;
51  while (i < pos - 1 && t != NULL) {
52  t = t->next;
53  i++;
54  }
55
```

```
56  if (t == NULL)
57  return;
58
59  n->next = t->next;
60  t->next = n;
61  }
62
63  void delete_begin() {
64  if (head == NULL)
65  return;
66  struct node *t = head;
67  head = head->next;
68  free(t);
69  }
70
71  void delete_end() {
72  if (head == NULL)
73  return;
74
75  if (head->next == NULL) {
76  free(head);
77  head = NULL;
78  return;
79  }
80
81  struct node *t = head;
82  while (t->next->next != NULL)
83  t = t->next;
84
```

```
85  free(t->next);
86  t->next = NULL;
87  }
88
89  void delete_pos() {
90  int pos, i = 1;
91  scanf("%d", &pos);
92
93  if (pos == 1) {
94  delete_begin();
95  return;
96  }
97
98  struct node *t = head;
99  while (i < pos - 1 && t != NULL) {
100 t = t->next;
101 i++;
102 }
103
104 if (t == NULL || t->next == NULL)
105 return;
106
107 struct node *d = t->next;
108 t->next = d->next;
109 free(d);
110 }
111
112 void display() {
113 struct node *t = head;
```

```
114 while (t != NULL) {
115 printf("%d ", t->data);
116 t = t->next;
117 }
118 printf("\n");
119 }
120
121 void modify_by_value() {
122 int oldv, newv;
123 scanf("%d %d", &oldv, &newv);
124
125 struct node *t = head;
126 while (t != NULL) {
127 if (t->data == oldv) {
128 t->data = newv;
129 return;
130 }
131 t = t->next;
132 }
133 }
134
135 void modify_by_pos() {
136 int pos, newv, i = 1;
137 scanf("%d %d", &pos, &newv);
138
139 struct node *t = head;
140 while (i < pos && t != NULL) {
141 t = t->next;
142 i++;
```

```
143 }
144
145 if (t != NULL)
146 t->data = newv;
147 }
148
149 int main() {
150 int ch;
151 while (1) {
152 scanf("%d", &ch);
153
154 if (ch == 1) insert_begin();
155 else if (ch == 2) insert_end();
156 else if (ch == 3) insert_pos();
157 else if (ch == 4) delete_begin();
158 else if (ch == 5) delete_end();
159 else if (ch == 6) delete_pos();
160 else if (ch == 7) display();
161 else if (ch == 8) modify_by_value();
162 else if (ch == 9) modify_by_pos();
163 else break;
164 }
165 return 0;
166 }
167
```

Output

```
1
10
2
20
2
30
7
10 20 30
8
20 25
7
10 25 30
9
2 99
7
10 99 30
4
7
99 30
0
```

Program 2: Search an Element in Singly Linked List

Program Code

2_search_sll.c

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

struct node {
int data;
struct node *next;
};

int main() {
struct node *head = NULL, *temp, *newnode;
int n, x, key, pos = 1, found = 0;

scanf("%d", &n);

for (int i = 0; i < n; i++) {
```

```
16  scanf("%d", &x);
17  newnode = malloc(sizeof(struct node));
18  newnode->data = x;
19  newnode->next = NULL;
20
21  if (head == NULL)
22  head = newnode;
23  else {
24  temp = head;
25  while (temp->next != NULL)
26  temp = temp->next;
27  temp->next = newnode;
28  }
29  }
30
31  scanf("%d", &key);
32
33  temp = head;
34  while (temp != NULL) {
35  if (temp->data == key) {
36  printf("Found at position %d", pos);
37  found = 1;
38  break;
39  }
40  temp = temp->next;
41  pos++;
42  }
43
44  if (!found)
```

```
45  printf("Not Found");
46
47  return 0;
48  }
49
```

Output

```
5
10 20 30 40 50
30
Found at position 3
```

Program 3: Count Number of Nodes in Singly Linked List

Program Code

3_count_nodes_sll.c

```
1   #include <stdio.h>
2   #include <stdlib.h>
3
4   struct node {
5   int data;
6   struct node *next;
7   };
8
9   int main() {
10  struct node *head = NULL, *temp, *newnode;
11  int n, x, count = 0;
12
13  scanf("%d", &n);
14
15  for (int i = 0; i < n; i++) {
16  scanf("%d", &x);
17  newnode = malloc(sizeof(struct node));
```

```
18  newnode->data = x;
19  newnode->next = NULL;
20
21  if (head == NULL)
22  head = newnode;
23  else {
24  temp = head;
25  while (temp->next != NULL)
26  temp = temp->next;
27  temp->next = newnode;
28  }
29  }
30
31  temp = head;
32  while (temp != NULL) {
33  count++;
34  temp = temp->next;
35  }
36
37  printf("Count = %d", count);
38
39  return 0;
40  }
41
```

Output

```
4
5 15 25 35
Count = 4
```

Program 4: Reverse a Singly Linked List

Program Code

4_reverse_sll.c

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

struct node {
int data;
struct node *next;
};

int main() {
struct node *head = NULL, *temp, *newnode;
struct node *prev = NULL, *curr, *next;
int n, x;

scanf("%d", &n);

for (int i = 0; i < n; i++) {
scanf("%d", &x);
newnode = malloc(sizeof(struct node));
newnode->data = x;
newnode->next = NULL;

if (head == NULL)
head = newnode;
else {
temp = head;
while (temp->next != NULL)
```

```
27  temp = temp->next;
28  temp->next = newnode;
29  }
30  }
31
32  curr = head;
33  while (curr != NULL) {
34  next = curr->next;
35  curr->next = prev;
36  prev = curr;
37  curr = next;
38  }
39
40  head = prev;
41
42  temp = head;
43  while (temp != NULL) {
44  printf("%d ", temp->data);
45  temp = temp->next;
46  }
47
48  return 0;
49  }
50
```

Output

```
6
1 2 3 4 5 6
6 5 4 3 2 1
```

Program 5: Sort a Singly Linked List

Program Code

5_sort_sll.c

```
1   #include <stdio.h>
2   #include <stdlib.h>
3
4   struct node {
5   int data;
6   struct node *next;
7   };
8
9   int main() {
10  struct node *head = NULL, *temp, *newnode, *i, *j;
11  int n, x, t;
12
13  scanf("%d", &n);
14
15  for (int k = 0; k < n; k++) {
16  scanf("%d", &x);
17  newnode = malloc(sizeof(struct node));
18  newnode->data = x;
19  newnode->next = NULL;
20
21  if (head == NULL)
22  head = newnode;
23  else {
24  temp = head;
25  while (temp->next != NULL)
26  temp = temp->next;
```

```
27  temp->next = newnode;
28  }
29  }
30
31  for (i = head; i != NULL; i = i->next) {
32  for (j = i->next; j != NULL; j = j->next) {
33  if (i->data > j->data) {
34  t = i->data;
35  i->data = j->data;
36  j->data = t;
37  }
38  }
39  }
40
41  temp = head;
42  while (temp != NULL) {
43  printf("%d ", temp->data);
44  temp = temp->next;
45  }
46
47  return 0;
48  }
49
```

Output

```
5
40 10 50 20 30
10 20 30 40 50
```

Result

All the programs related to Singly Linked List were successfully implemented and executed.