



main.c

Output



```
1  #include <stdio.h>
2  #include <stdlib.h>
3  #include <string.h>
4  struct Feedback {
5      int studentID;
6      char courseCode[20];
7      int rating;
8      char comments[100];
9      struct Feedback* next;
10 };
11 struct Feedback* head = NULL;
12 struct Feedback* createNode(int id,
    char course[], int rating, char
    comments[]) {
13     struct Feedback* newNode =
        (struct Feedback*)malloc
        (sizeof(struct Feedback));
14     newNode->studentID = id;
15     strcpy(newNode->courseCode,
        course);
16     newNode->rating = rating;
17     strcpy(newNode->comments,
        comments);
18     newNode->next = NULL;
19     return newNode;
20 }
21 void addFeedback(int id, char
    course[], int rating, cha
```

Run





main.c

Output



```
        course[], int rating, char
        comments[]) {
22      struct Feedback* newNode =
           createNode(id, course, rating
           , comments);
23      if (head == NULL) {
24          head = newNode;
25      } else {
26          struct Feedback* temp = head;
27          while (temp->next != NULL)
28              temp = temp->next;
29          temp->next = newNode;
30      }
31      printf("Feedback added
           successfully!\n");
32  }
33  void displayFeedback() {
34      if (head == NULL) {
35          printf("No feedback records
           available.\n");
36          return;
37      }
38      struct Feedback* temp = head;
39      printf("\n---- Feedback Records
           ----\n");
40      while (temp != NULL) {
41          printf("Student ID: %d
           Course: %s, Rating: %.2f",
           temp->id, temp->course, temp->rating);
```

Run



main.c

Output



```
Course: %s, Rating: %d,
Comments: %s\n",
42         temp->studentID, temp
           ->courseCode, temp
           ->rating, temp
           ->comments);
43         temp = temp->next;
44     }
45 }
46 void displayReverse(struct Feedback*
    node) {
47     if (node == NULL) return;
48     displayReverse(node->next);
49     printf("Student ID: %d, Course:
        %s, Rating: %d, Comments:
        %s\n",
50         node->studentID, node
           ->courseCode, node
           ->rating, node
           ->comments);
51 }
52 void searchByStudentID(int id) {
53     struct Feedback* temp = head;
54     int found = 0;
55     while (temp != NULL) {
56         if (temp->studentID == id) {
57             printf("Found ->
                ID: %d, Course: %s,
```

Run





main.c

Output



```
Rating: %d, Comments:
%s\n",
58         temp->studentID,
           temp->courseCode,
           temp->rating, temp
           ->comments);
59         found = 1;
60     }
61     temp = temp->next;
62 }
63 if (!found) printf("No feedback
           found for Student ID %d\n",
           id);
64 }
65 void searchByCourse(char course[]) {
66     struct Feedback* temp = head;
67     int found = 0;
68     while (temp != NULL) {
69         if (strcmp(temp->courseCode,
           course) == 0) {
70             printf("Found -> Student
           ID: %d, Course: %s,
           Rating: %d, Comments:
           %s\n",
71                 temp->studentID,
           temp->courseCode,
           temp->rating,
           temp->comments),
```

Run



main.c

Output



```
72         found = 1;
73     }
74     temp = temp->next;
75 }
76 if (!found) printf("No feedback
              found for course %s\n",
              course);
77 }
78 void averageRating(char course[]) {
79     struct Feedback* temp = head;
80     int sum = 0, count = 0;
81     while (temp != NULL) {
82         if (strcmp(temp->courseCode,
83                   course) == 0) {
84             sum += temp->rating;
85             count++;
86         }
87         temp = temp->next;
88     }
89     if (count == 0) {
90         printf("No feedback for
91               course %s\n", course);
92     } else {
93         printf("Average rating for %s
94               : %.2f\n", course, (float)
95               sum / count);
96     }
97 }
```

Run





main.c

Output



```
94 struct Feedback* cloneList(struct
    Feedback* head) {
95     if (head == NULL) return NULL;
96     struct Feedback* newHead = NULL,
        *tail = NULL;
97     struct Feedback* temp = head;
98     while (temp != NULL) {
99         struct Feedback* newNode =
            createNode(temp
                ->studentID, temp
                ->courseCode, temp
                ->rating, temp->comments
            );
100         if (newHead == NULL) {
101             newHead = newNode;
102             tail = newNode;
103         } else {
104             tail->next = newNode;
105             tail = newNode;
106         }
107         temp = temp->next;
108     }
109     return newHead;
110 }
111
112 int main() {
113     int choice, id, rating;
114     char course[20], comments[100];
```

Run





main.c

Output



```
115     struct Feedback* clonedList =
        NULL;

116
117     while (1) {
118         printf("\n--- Student
            Feedback Tracking System
            ---\n");

119         printf("1. Add Feedback\n");
120         printf("2. Search by Student
            ID\n");

121         printf("3. Search by Course
            Code\n");

122         printf("4. Calculate Average
            Rating (Course Wise)\n");
123         printf("5. Display All
            Feedback\n");
124         printf("6. Display Feedback
            in Reverse\n");

125         printf("7. Clone Feedback
            Data\n");

126         printf("8. Exit\n");
127         printf("Enter your choice: "
            );
128         scanf("%d", &choice);
129
130         switch (choice) {
131             case 1:
132                 printf("Enter Student
```

Run





main.c

Output



```
133 scanf("%d", &id);
134 printf("Enter Course
    Code: ");
135 scanf("%s", course);
136 printf("Enter Rating
    (1-5): ");
137 scanf("%d", &rating);
138 getchar();
139 printf("Enter
    Comments: ");
140 fgets(comments,
    sizeof(comments),
    stdin);
141 comments[strcspn
    (comments, "\n")] =
    '\0'; |
142 addFeedback(id,
    course, rating,
    comments);
143 break;
144
145 case 2:
146     printf("Enter Student
    ID to search: ");
147 scanf("%d", &id);
148 searchByStudentID(id
    );
149 break;
```

Run



main.c

Output



```
151         case 3:
152             printf("Enter Course
153                 Code to search: ");
154             scanf("%s", course);
155             searchByCourse(course
156                 );
157             break;
158         case 4:
159             printf("Enter Course
160                 Code to calculate
161                 average rating: ");
162             scanf("%s", course);
163             averageRating(course
164                 );
165             break;
166         case 5:
167             displayFeedback();
168             break;
169         case 6:
170             if (head == NULL) {
171                 printf("No
172                     feedback records
173                     .\n");
174             } else {
175                 printf("\n
```

Run



main.c

Output



```
Feedback in Reverse
----\n");
172         displayReverse
        (head);
173     }
174     break;
175
176     case 7:
177         clonedList =
            cloneList(head);
178         printf("Feedback list
            cloned
            successfully!\n");
179         break;
180
181     case 8:
182         printf("Exiting...\n"
            );
183         exit(0);
184
185     default:
186         printf("Invalid
            choice! Try again
            .\n");
187     }
188 }
189 return 0;
190 }
```

Run





main.c

Output



```
--- Student Feedback Tracking System ---
```

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

```
Enter your choice: 1
```

```
Enter Student ID: 100
```

```
Enter Course Code: AIML
```

```
Enter Rating (1-5): 4
```

```
Enter Comments: GOOD TEACHING
```

```
Feedback added successfully!
```

```
--- Student Feedback Tracking System ---
```

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

```
Enter your choice: |
```



main.c

Output



--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: 2

Enter Student ID to search: 100

Found -> Student ID: 100, Course: AIML,
Rating: 4, Comments: GOOD TEACHING

--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: 3

Enter Course Code to search: AIML

Found -> Student ID: 100, Course: AIML,
Rating: 4, Comments: GOOD TEACHING



main.c

Output



Rating: 4, Comments: GOOD TEACHING

--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: 4

Enter Course Code to calculate average
rating: AIML

Average rating for AIML: 4.00

--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: 5

---- Feedback Records ----

Student ID: 100 Course: AIML Rating: 4



main.c

Output



rating: AIML

Average rating for AIML: 4.00

--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: 5

---- Feedback Records ----

Student ID: 100, Course: AIML, Rating: 4,
Comments: GOOD TEACHING

--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: |



main.c

Output



Comments: GOOD TEACHING

--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: 6

---- Feedback in Reverse ----

Student ID: 100, Course: AIML, Rating: 4,
Comments: GOOD TEACHING

--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: 7

Feedback list cloned successfully!



main.c

Output



Comments: GOOD TEACHING

--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: 7

Feedback list cloned successfully!

--- Student Feedback Tracking System ---

1. Add Feedback
2. Search by Student ID
3. Search by Course Code
4. Calculate Average Rating (Course Wise)
5. Display All Feedback
6. Display Feedback in Reverse
7. Clone Feedback Data
8. Exit

Enter your choice: 8

Exiting...

=== Code Execution Successful ===