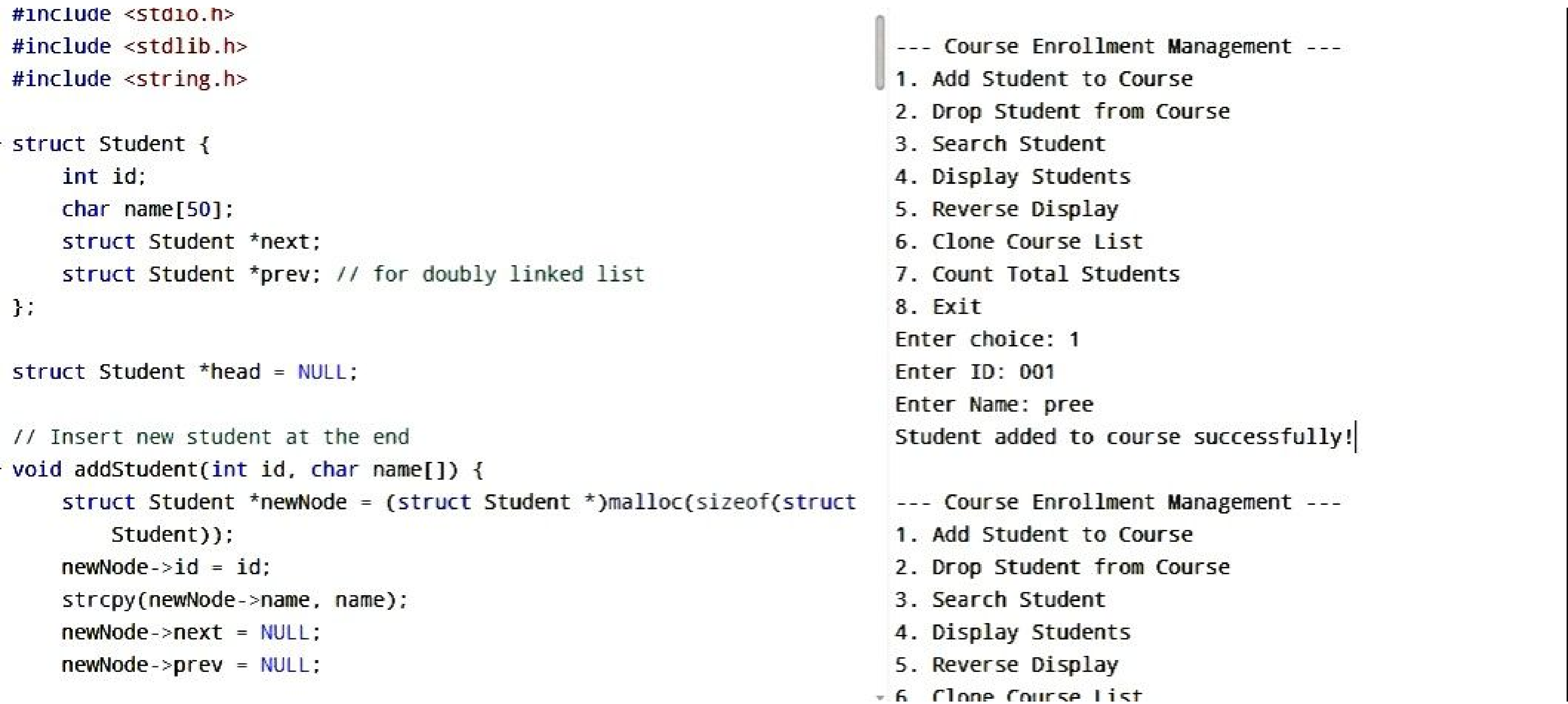
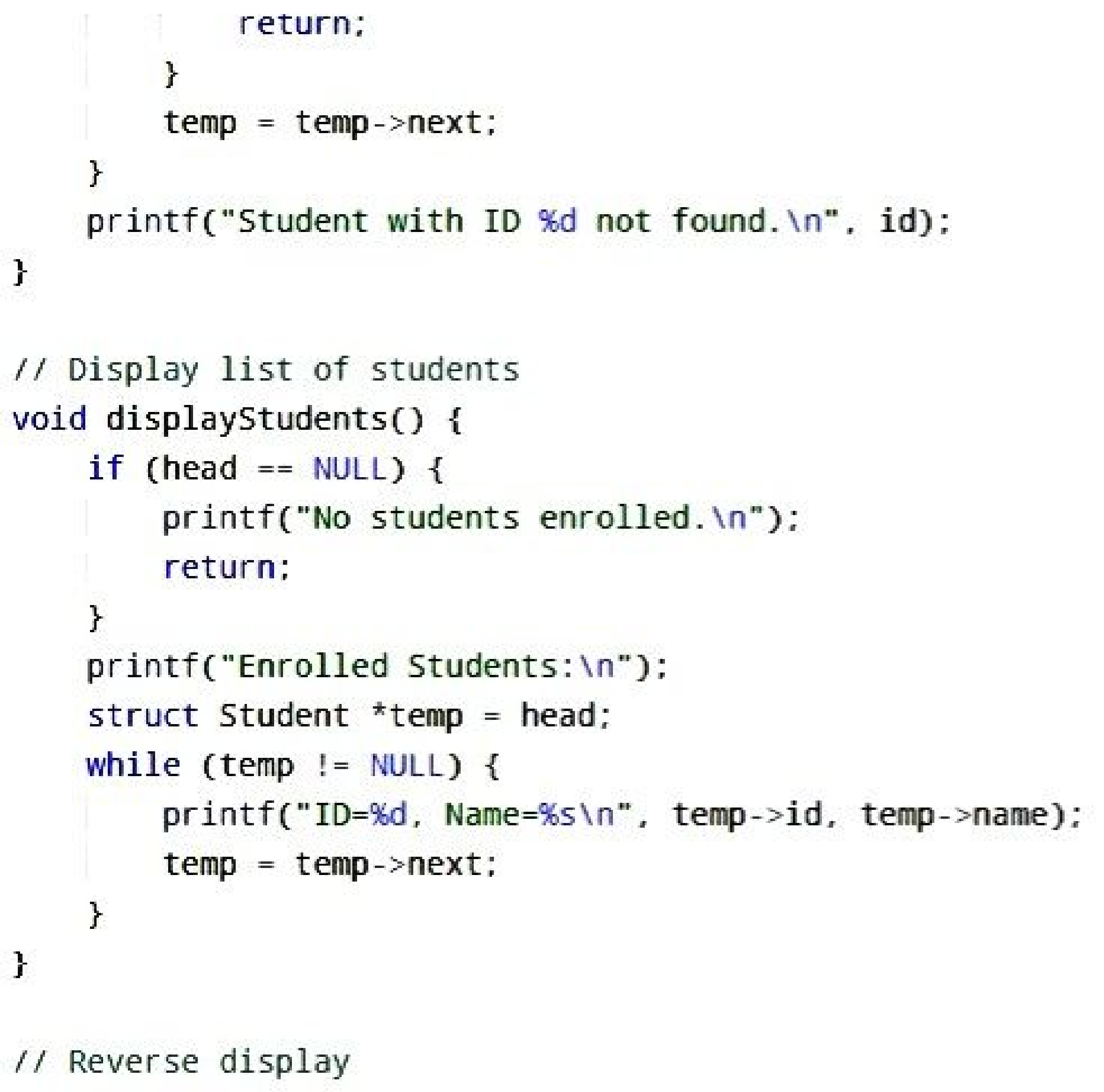
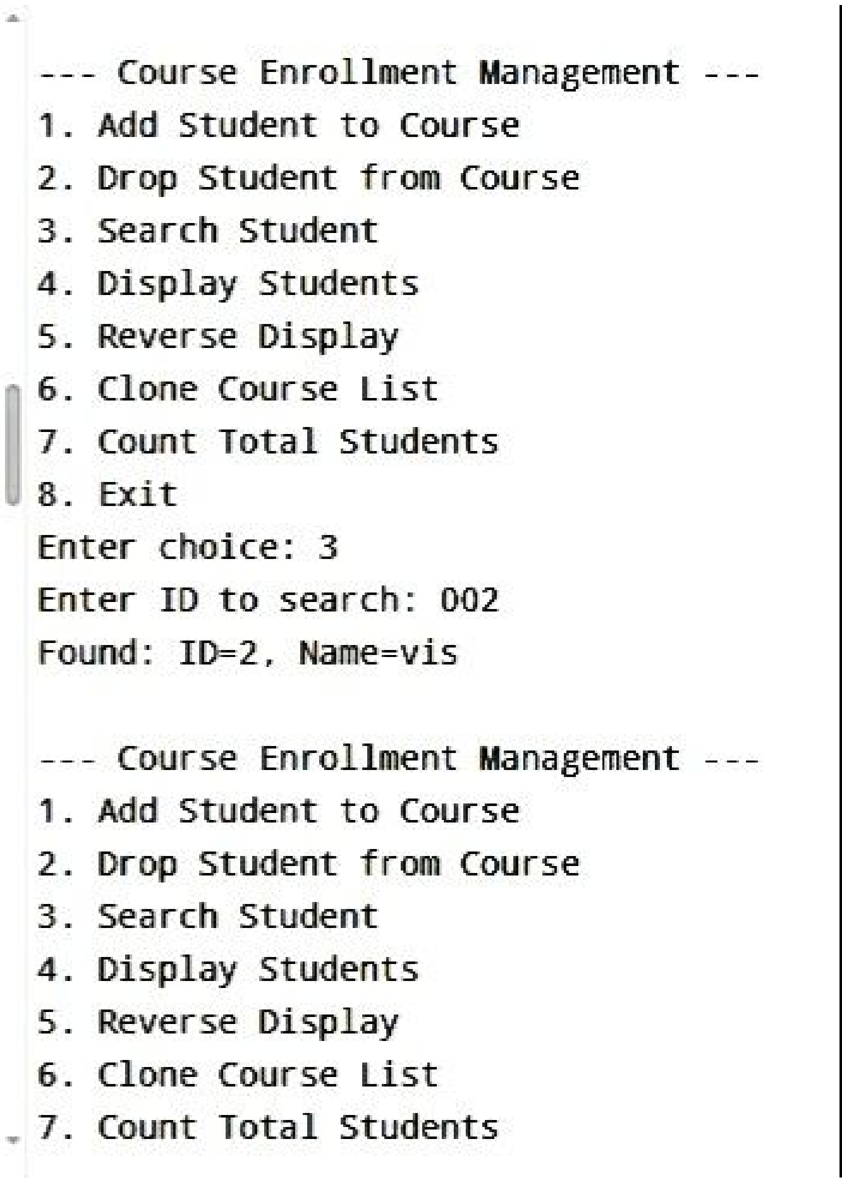


```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Student {
    int id;
    char name[50];
    struct Student *next;
    struct Student *prev; // for doubly linked list
};

struct Student *head = NULL;

// Insert new student at the end
void addStudent(int id, char name[]) {
    struct Student *newNode = (struct Student *)malloc(sizeof(struct
        Student));
    newNode->id = id;
    strcpy(newNode->name, name);
    newNode->next = NULL;
    newNode->prev = NULL;
```

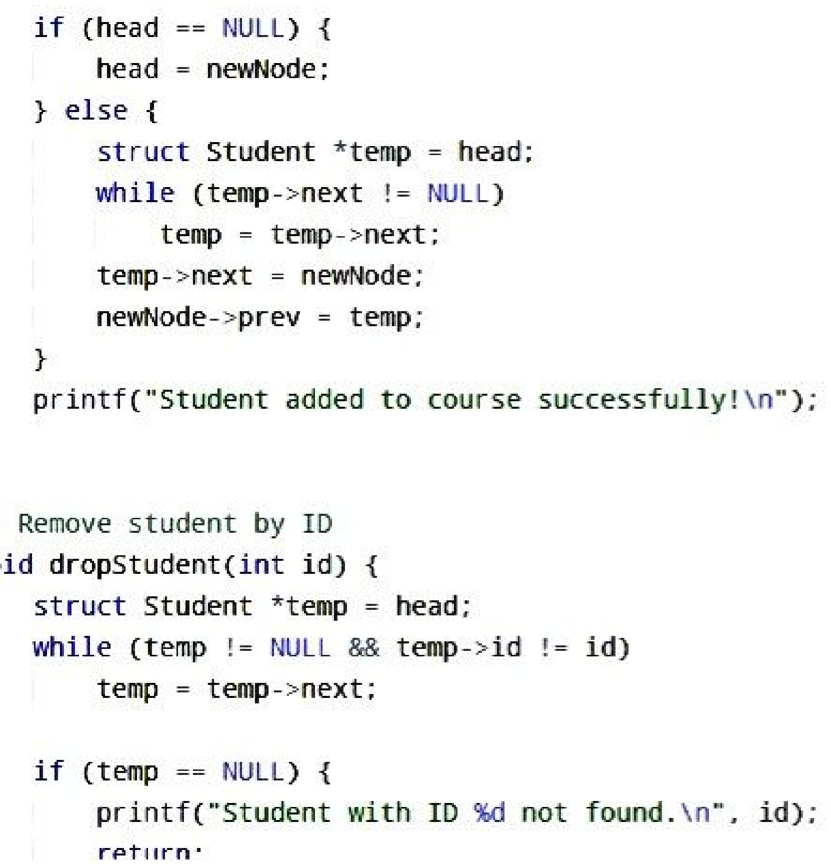
```
--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 1
Enter ID: 001
Enter Name: pree
Student added to course successfully!

--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
```



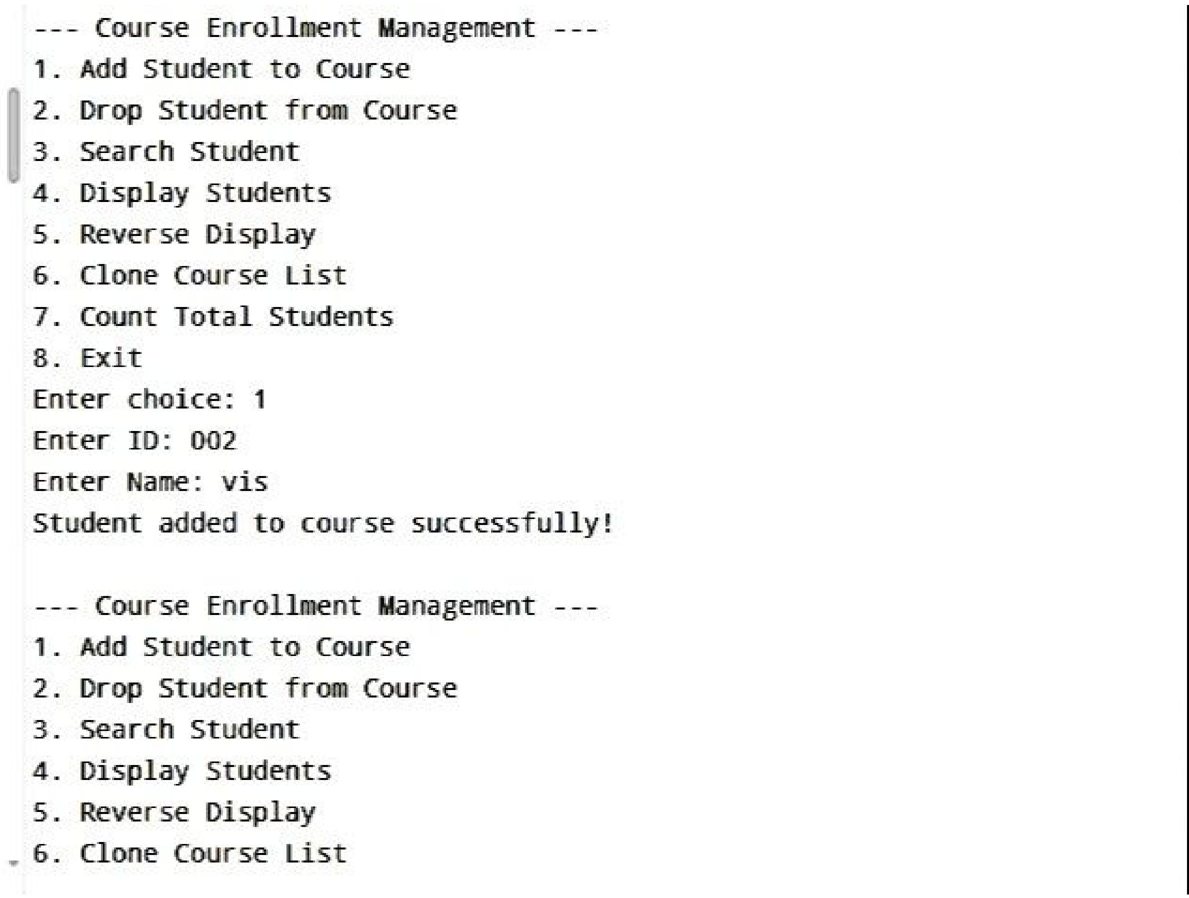
```c
            return;
        }
        temp = temp->next;
    }
    printf("Student with ID %d not found.\n", id);
}

// Display list of students
void displayStudents() {
    if (head == NULL) {
        printf("No students enrolled.\n");
        return;
    }
    printf("Enrolled Students:\n");
    struct Student *temp = head;
    while (temp != NULL) {
        printf("ID=%d, Name=%s\n", temp->id, temp->name);
        temp = temp->next;
    }
}

// Reverse display
```



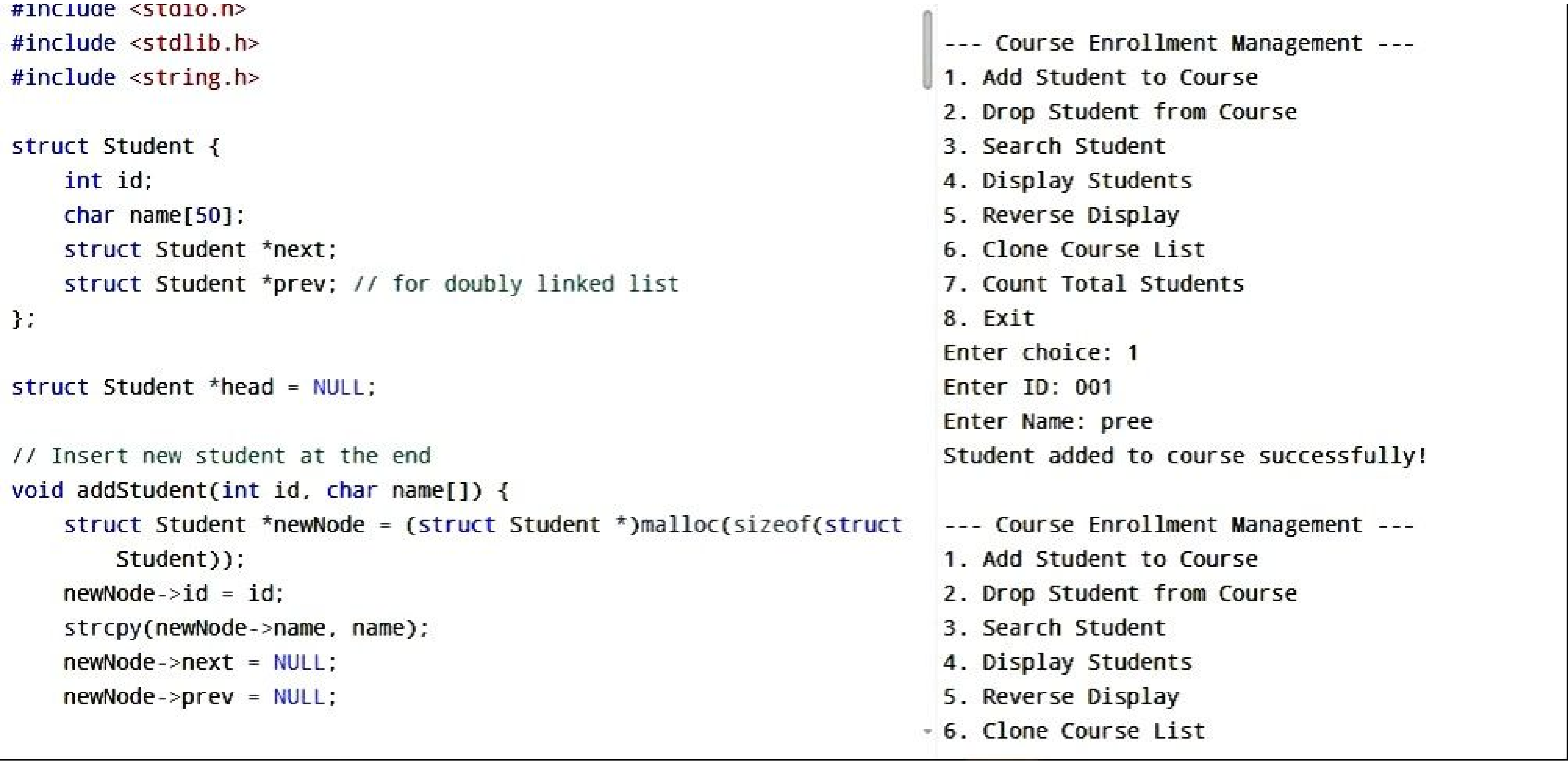
```
--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 3
Enter ID to search: 002
Found: ID=2, Name=vis

--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
```



```c
  if (head == NULL) {
      head = newNode;
  } else {
      struct Student *temp = head;
      while (temp->next != NULL)
          temp = temp->next;
      temp->next = newNode;
      newNode->prev = temp;
  }
  printf("Student added to course successfully!\n");


 Remove student by ID
id dropStudent(int id) {
  struct Student *temp = head;
  while (temp != NULL && temp->id != id)
      temp = temp->next;

  if (temp == NULL) {
      printf("Student with ID %d not found.\n", id);
      return;
```

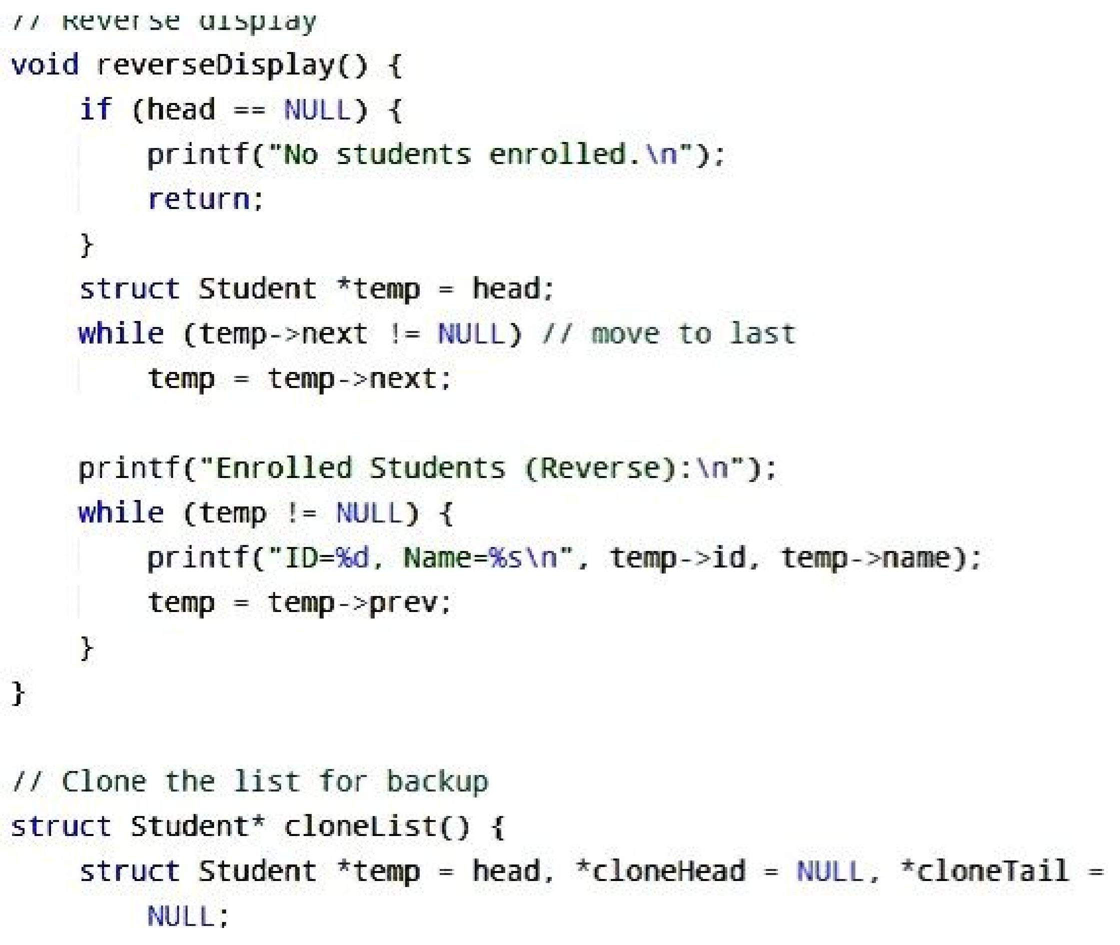


```
--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 1
Enter ID: 002
Enter Name: vis
Student added to course successfully!

--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
```

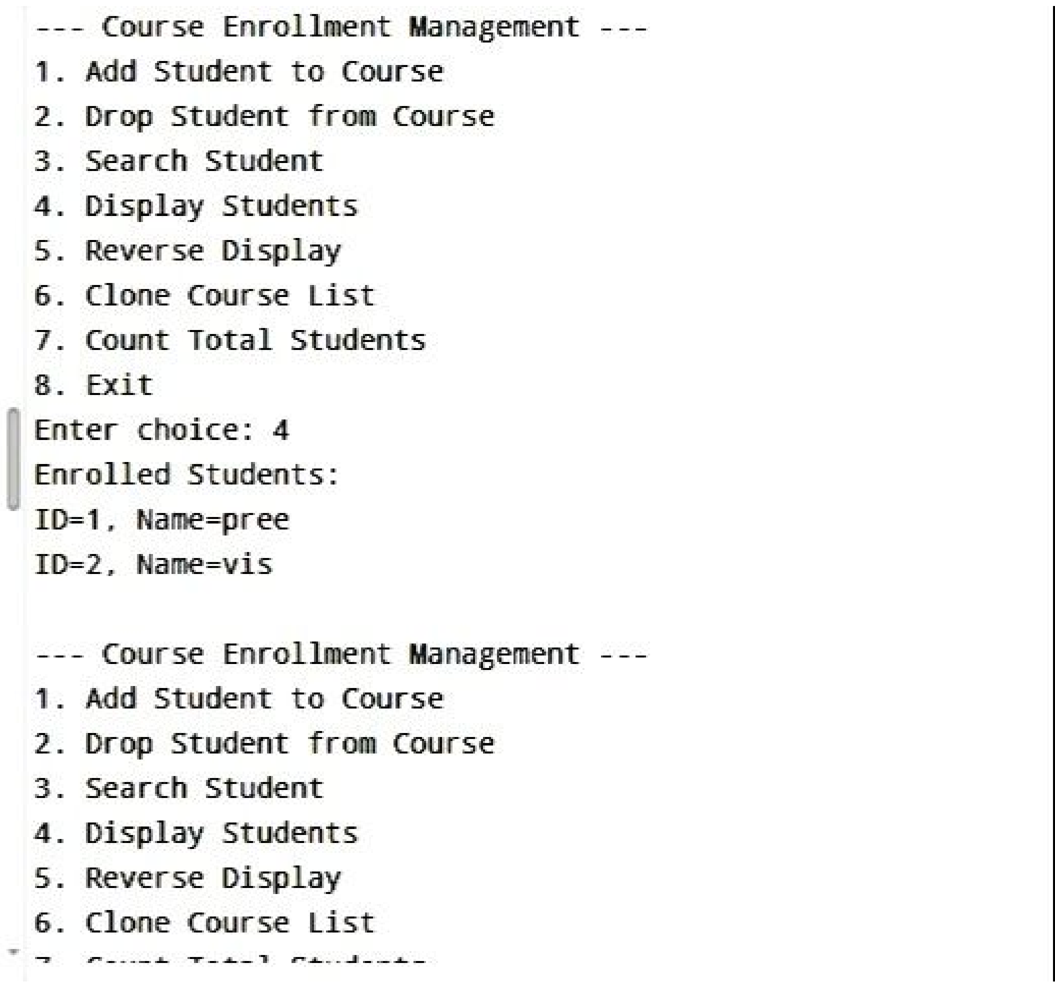


```c
#include <stdio.h>
#include <stdlib.h>
#include <string.h>

struct Student {
    int id;
    char name[50];
    struct Student *next;
    struct Student *prev; // for doubly linked list
};

struct Student *head = NULL;

// Insert new student at the end
void addStudent(int id, char name[]) {
    struct Student *newNode = (struct Student *)malloc(sizeof(struct
        Student));
    newNode->id = id;
    strcpy(newNode->name, name);
    newNode->next = NULL;
    newNode->prev = NULL;
```

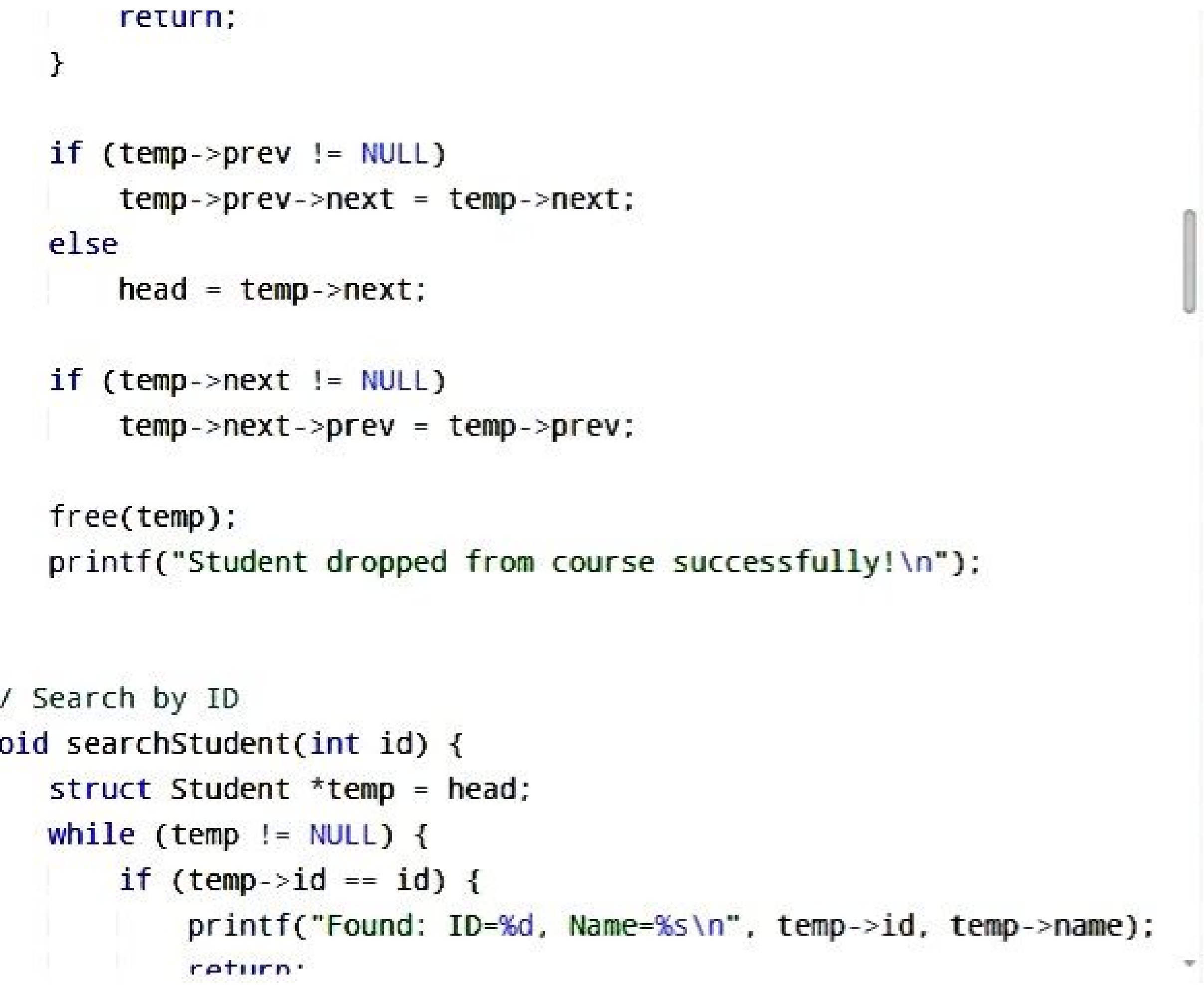
```
--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 1
Enter ID: 001
Enter Name: pree
Student added to course successfully!

--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
```



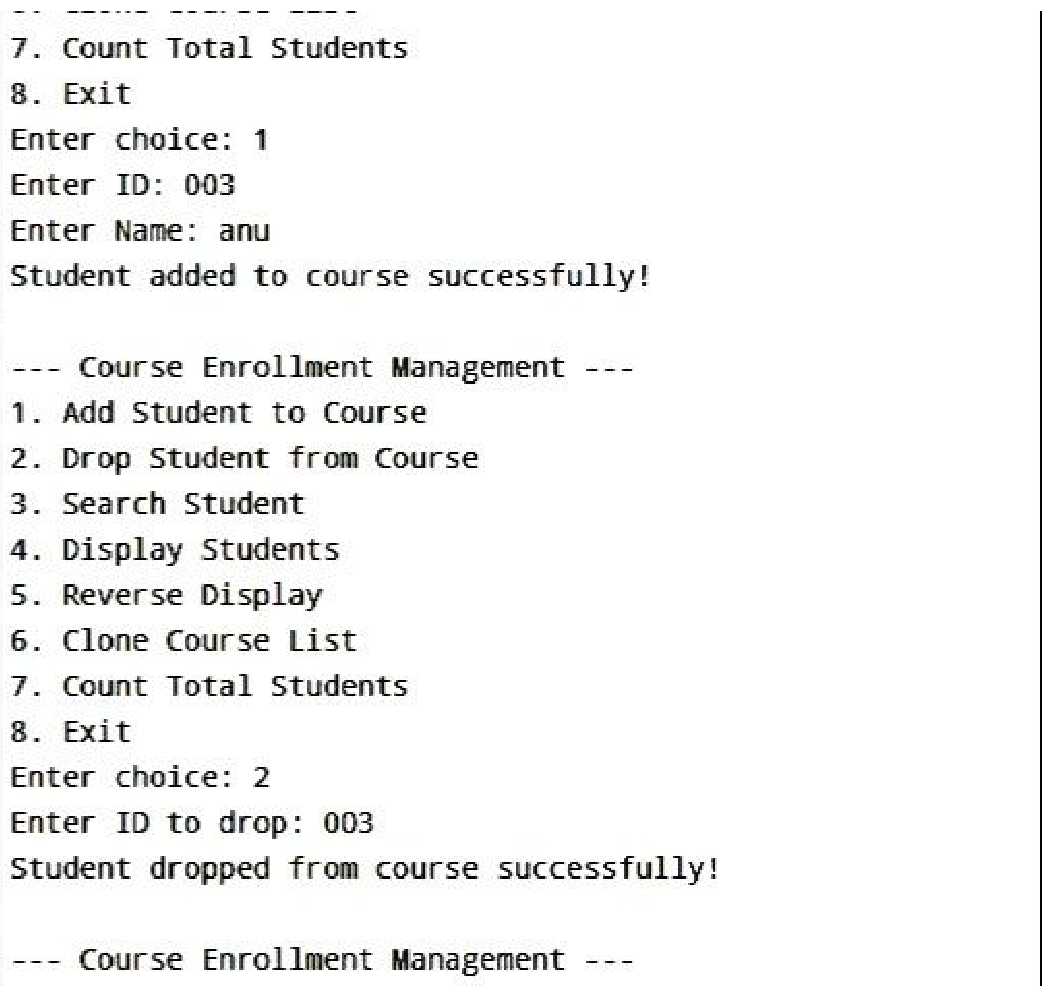
```c
// Reverse display
void reverseDisplay() {
    if (head == NULL) {
        printf("No students enrolled.\n");
        return;
    }
    struct Student *temp = head;
    while (temp->next != NULL) // move to last
        temp = temp->next;

    printf("Enrolled Students (Reverse):\n");
    while (temp != NULL) {
        printf("ID=%d, Name=%s\n", temp->id, temp->name);
        temp = temp->prev;
    }
}

// Clone the list for backup
struct Student* cloneList() {
    struct Student *temp = head, *cloneHead = NULL, *cloneTail =
        NULL;
```



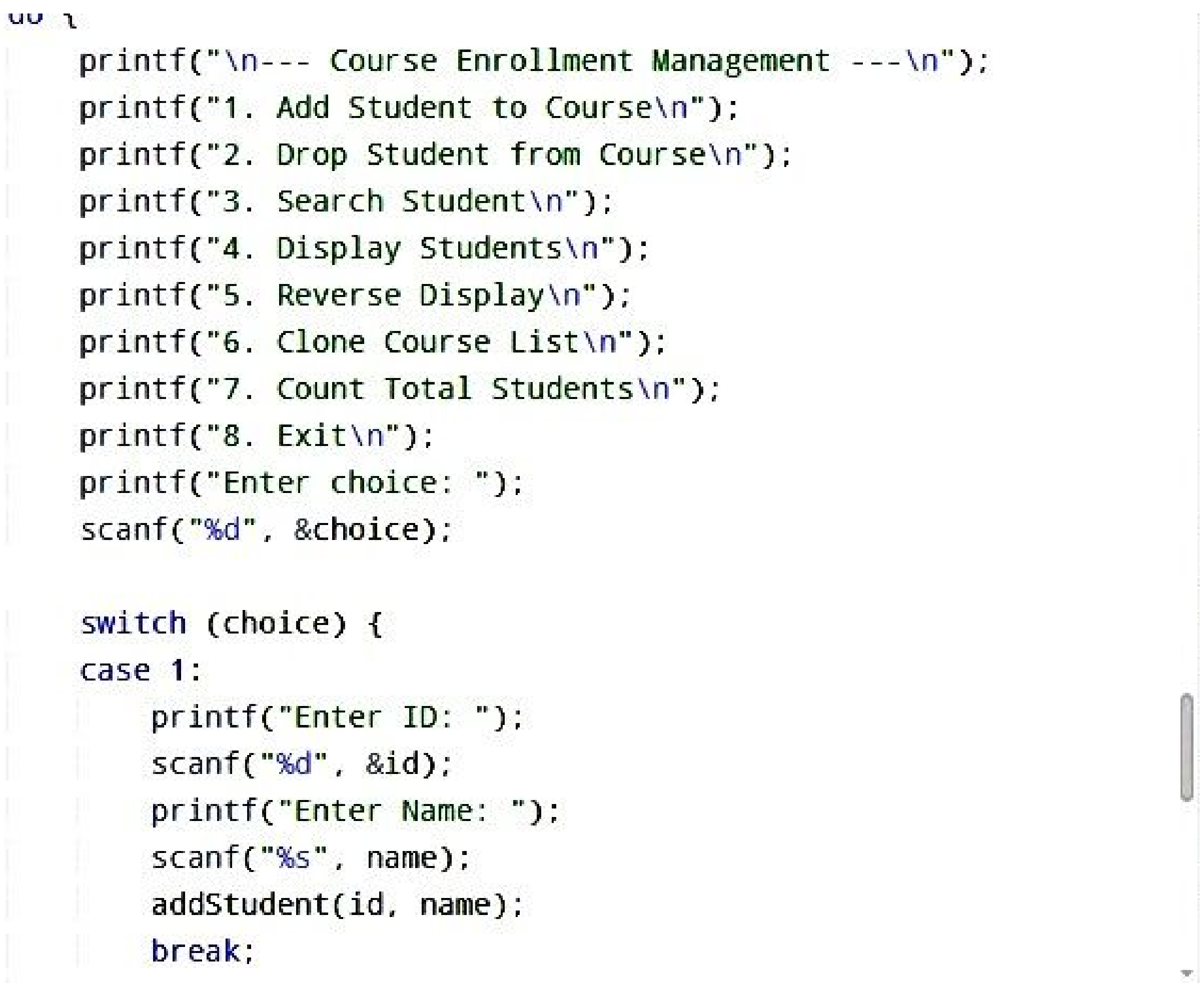
```
--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 4
Enrolled Students:
ID=1, Name=pree
ID=2, Name=vis

--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
```



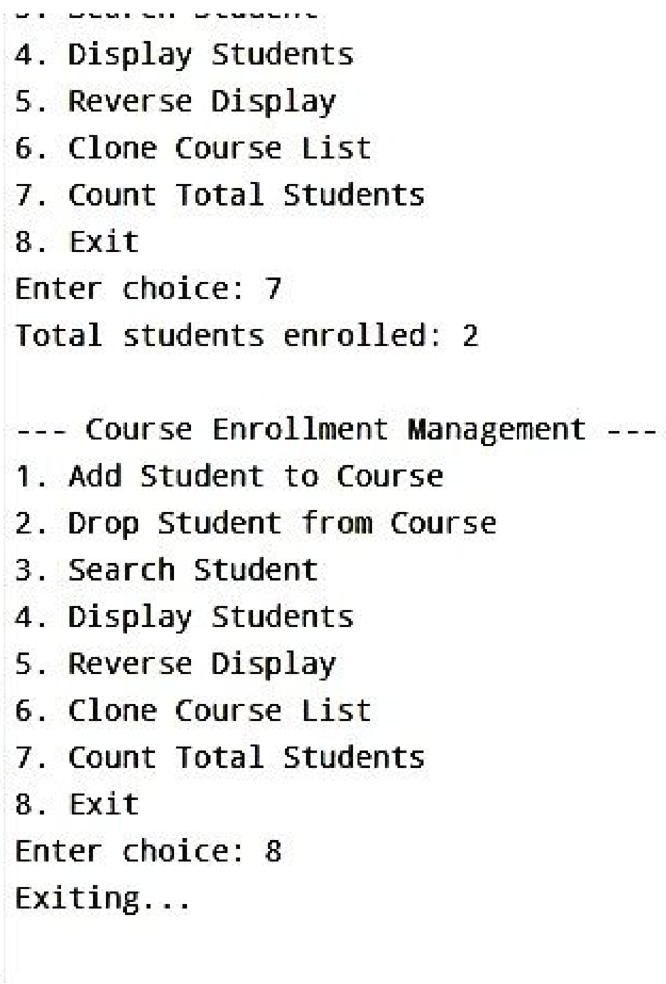
```
    return;
}

if (temp->prev != NULL)
    temp->prev->next = temp->next;
else
    head = temp->next;

if (temp->next != NULL)
    temp->next->prev = temp->prev;

free(temp);
printf("Student dropped from course successfully!\n");


/ Search by ID
oid searchStudent(int id) {
  struct Student *temp = head;
  while (temp != NULL) {
      if (temp->id == id) {
          printf("Found: ID=%d, Name=%s\n", temp->id, temp->name);
          return;
```



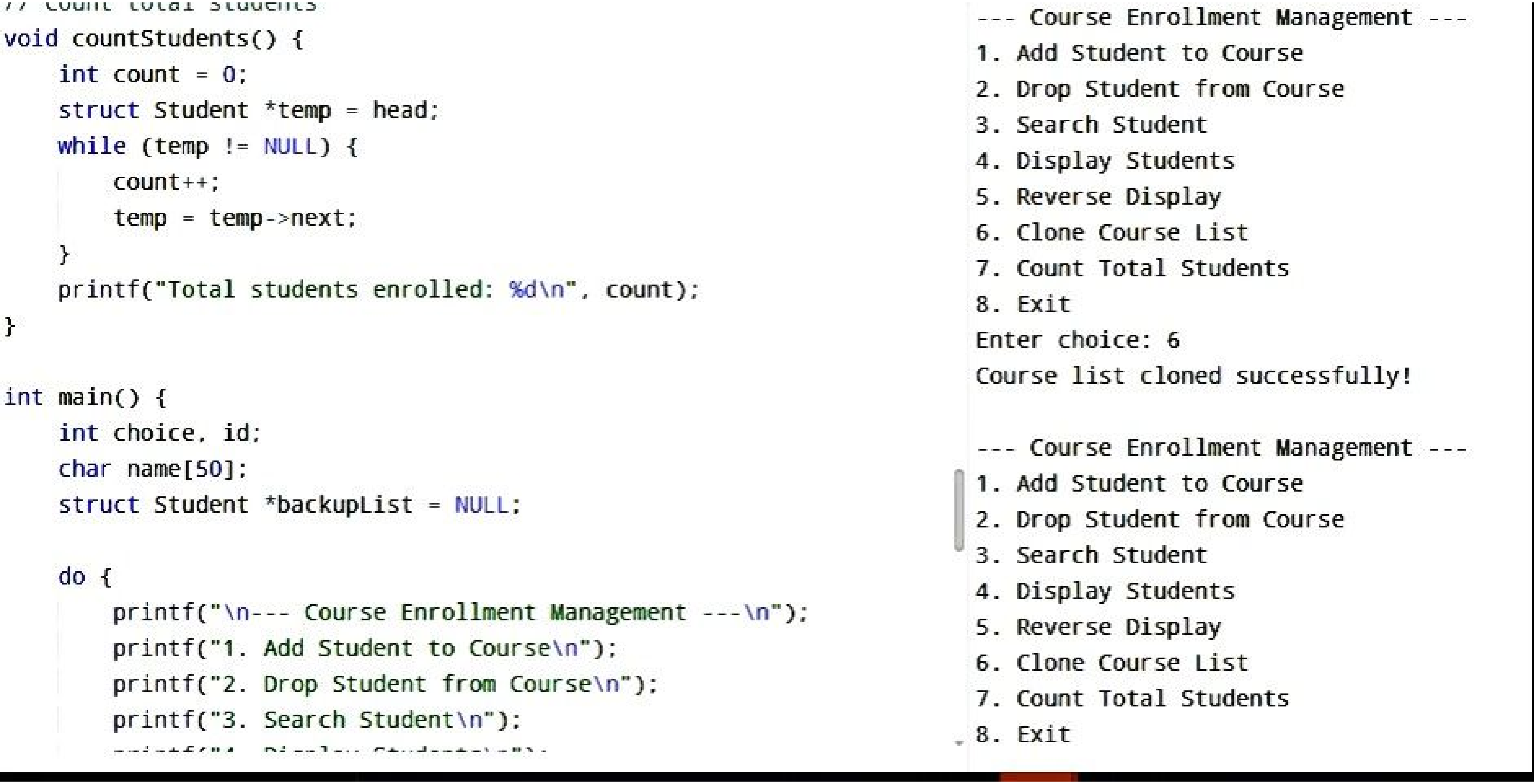
```
7. Count Total Students
8. Exit
Enter choice: 1
Enter ID: 003
Enter Name: anu
Student added to course successfully!

--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 2
Enter ID to drop: 003
Student dropped from course successfully!

--- Course Enrollment Management ---
```



```
do {
    printf("\n--- Course Enrollment Management ---\n");
    printf("1. Add Student to Course\n");
    printf("2. Drop Student from Course\n");
    printf("3. Search Student\n");
    printf("4. Display Students\n");
    printf("5. Reverse Display\n");
    printf("6. Clone Course List\n");
    printf("7. Count Total Students\n");
    printf("8. Exit\n");
    printf("Enter choice: ");
    scanf("%d", &choice);

    switch (choice) {
    case 1:
        printf("Enter ID: ");
        scanf("%d", &id);
        printf("Enter Name: ");
        scanf("%s", name);
        addStudent(id, name);
        break;
```

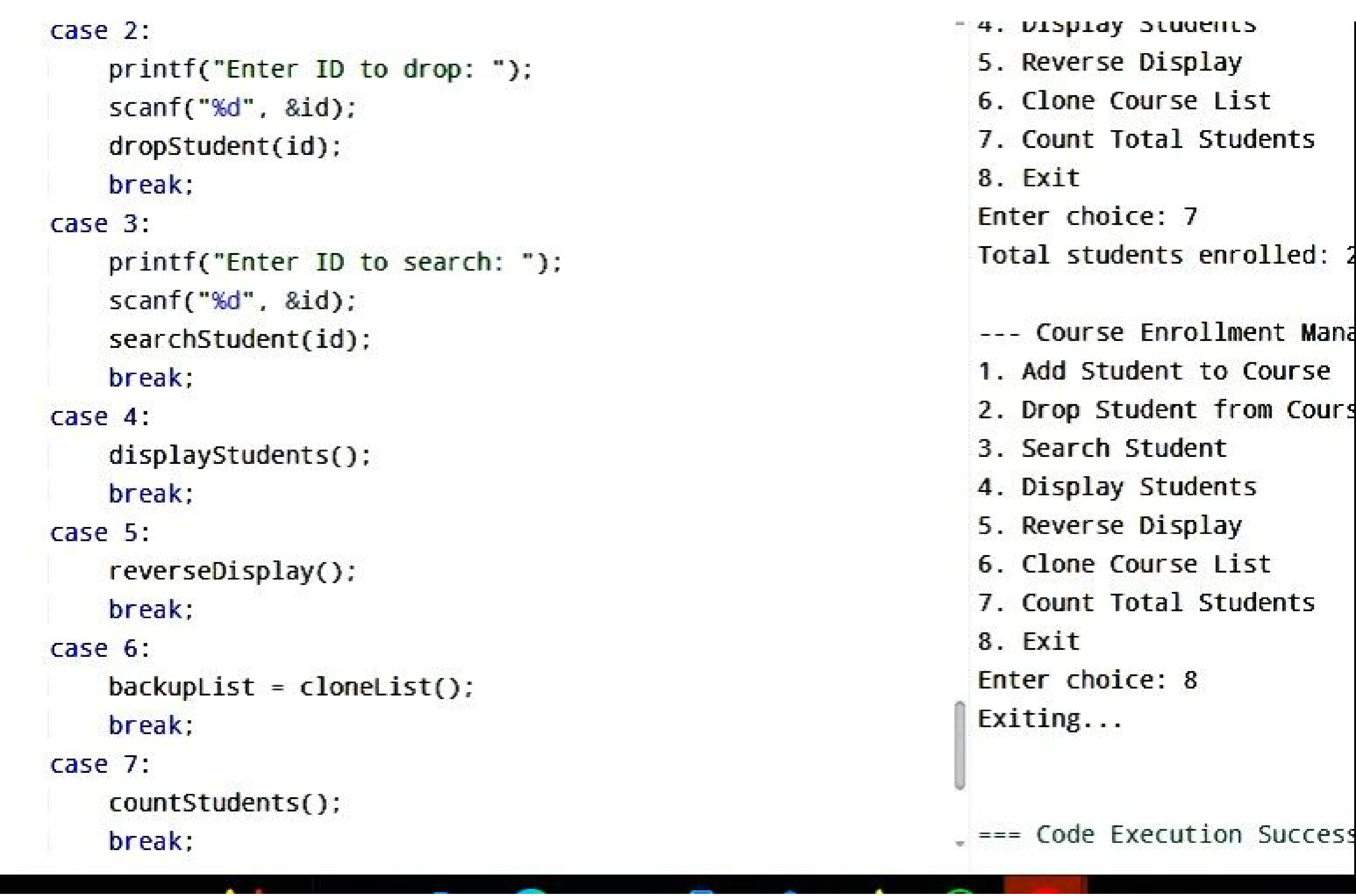


```
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 7
Total students enrolled: 2

--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 8
Exiting...
```



```c
// Count total students
void countStudents() {
    int count = 0;
    struct Student *temp = head;
    while (temp != NULL) {
        count++;
        temp = temp->next;
    }
    printf("Total students enrolled: %d\n", count);
}

int main() {
    int choice, id;
    char name[50];
    struct Student *backupList = NULL;

    do {
        printf("\n--- Course Enrollment Management ---\n");
        printf("1. Add Student to Course\n");
        printf("2. Drop Student from Course\n");
        printf("3. Search Student\n");
        printf("4. Display Students\n");
```

```
--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 6
Course list cloned successfully!

--- Course Enrollment Management ---
1. Add Student to Course
2. Drop Student from Course
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
```



```
case 2:
    printf("Enter ID to drop: ");
    scanf("%d", &id);
    dropStudent(id);
    break;
case 3:
    printf("Enter ID to search: ");
    scanf("%d", &id);
    searchStudent(id);
    break;
case 4:
    displayStudents();
    break;
case 5:
    reverseDisplay();
    break;
case 6:
    backupList = cloneList();
    break;
case 7:
    countStudents();
    break;
```

```
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 7
Total students enrolled: 2

--- Course Enrollment Mana
1. Add Student to Course
2. Drop Student from Cours
3. Search Student
4. Display Students
5. Reverse Display
6. Clone Course List
7. Count Total Students
8. Exit
Enter choice: 8
Exiting...


=== Code Execution Success
```