```
Student hostel Allotment System.
```

```
# include 2 stdio. hs
# include LSHdlib. hs
# include Zstring. hs
typedet struct student {
     Char studentID[20];
     Chan name [50];
      int poomNo;
      Char block [10];
       struct Student " hext;
 3 Student;
Student " head = NULL;
 Student " createwoode [ chan id [], Chan name [], int noom.
                              chan block[])}
     Student neurode: (Student*) malloc(size of (student));
      strupy (new Node -> student ID, id);
      Stripy (new Node -) name, hame);
      newNode -> nomNo = nom;
       Stripy (new Node -> block, block);
       newNode - next = NULL;
       return newNode;
  3
 void add Albiment (then id[], then name [], int wom, charblou[])
£.
   Student * hew Node = create Node (id, name, noom, block);
```

```
if (head = = NULL) (
      head . new Node;
3 else &
    student temp : head;
    while ( temp -> next ! = NULL)
         temp: temp - next;
        temp = next = new Node;
    printf ("Albtmeht added successfully!\n");
3
void removestudent (chan id[]) {
     if (head == NULL) {
         prints ("No necords found. \n");
         return;
      3
      Student "temp: head;
      student prer = NULL;
      while I temp != NULL && stremp (temp - studentio, id) != 0)
      pre = temp;
       temp : temp - next;
    15(timp = = NULL) {
        prints ( student 10 4.5 not found . \n", id);
         netun;
     if (prer = = NULL) {
          head : head > next;
    I else &
```

```
prev -> next : temp -> next;
    free(temp);
     print (" Student with 10 4.5 sonoved successfully. \n", id);
   3
void search By Name (char name []) {
    Student * temp: head;
    int found : 0;
     while (temp! = NULL) {
         if (strump (temp + name, name) = = 0) {
             prints (" found : 10 = 1.5, Name = 1.5, Room = 1.d,
                     Block = 1.5 \n,"
                  temp - studento, temp - name, temp - room NO,
                  temp -> block);
                 found = 1;
             temp = temp = next;
          if (I found) prints ('po student with name 1.5 found.
                                             (n', name);
          3
Yord Search By Room (in 1 room) {
     Student temp: head;
      int found = 0;
      while (temp 1 - NULL) 5
          if (temp - room No = = room) {
              printf ( "found: so = 1.5, Name = 1.5, Room = 1.d.
                              Block = 1.5 (n",
```

```
ferry > Studen+1D, temp > name, temp > roomNo, temp > block);
   found : 1;
   temp : temp = next;
  it (! found) prints ("no student found in room id. \", room);
 3
void display block Wise (chan block []) ?
      Student * temp = head ;
       in+ found = 0;
       printf (" Students in Block y.s: \n", block);
       while (temp ! = NULL) {
           if (Atrump (temp -> block, block) == 0) {
               printf ( ID= 1.5, Name = 1.5, Room = 1.al In",
                temp - student 10, temp - name, temp - noom No);
                found = 1;
               temp = temp - nex +;
             3
               if [I found ) points (" No students in block 1.5. 10", block
             3
 void neverse Display (student * node) &
          if (node = = NULL) neturn;
           neverse Display (node - next);
           prints ("ID = 1.5, Name = 1.5, Room = 1.d, Block = 1.5 \n",
           node + student 10, node - name, node - room No, node + block)
```

3

```
Student " Mondist ( Student " head ) {
    if ( head == NULL) neturn NULL;
    student " new Head : NULL, * tail = NULL;
    student & temp : head ;
      while (temp! : NULL) {
          Student " new Node = createNode (temp -) student 1D,
              temp-s name, temp + noom No, temp + block);
          if (new Head = = NULL) {
              nuvHead = new Node;
              fail = new Head;
          3 else 1
             tail - next = new Node;
              tail = new Noale;
           temp = temp = next;
        neturn newHead;
    3
void count PerBlock() {
     student " temp : head;
     int count A = 0, count B = 0, count (=0, bount Other = 0;
     while (temp! = NULL) {
          if (strump (temp => blowr, "A") == 0) count A++;
          else if (stremp (temp > block = "B") == 0) Lount B++;
          else if (stromp (temp + block = "(") = = 0) (ount C++;
          else contatother ++;
```

```
temp : temp - next;
3
   printf (" Block A: " of students In ", wunts);
   printf ("Block 5: Y.d students \n", counts);
   printf ("Block (: 1.d students \n", count ();
    printf ("Other Bloves; 1.d Students In", count other);
 3
int main() {
    int Choice, room;
    char id [20], name [50], blove [10].
    Student & Clonelist: NULL;
    while (1) 5
      prints ("In --- Student Hostel Albetment System --- In");
       printf ("1. Add new Allotment In");
       printf ("2. Remove student (n");
       printf ("3. Search by Name In");
        printf (" 4. Search by Name (n");
         printf (" 5, Display Allotments Block - wise in");
         printf ("i. Reverse Display \n");
          printf (" 1, clone List In");
           printf ("8. Court Students per Block (n");
           printf (" 9. Exit (n");
            printf (" Enten your choice;");
            Stanf (" 1.d", 2 choice);
  Switch (dwile) {
```

```
case 1:
   print f (" Enter Student 10: ");
   scanf (" 1.d [1/n]", id);
    printf (" Enter Name: ");
    scanf ("1. [^\n]", rame);
    Prints (" Enter Room Number:");
     scanf (" 1. d", & room);
     printf (" Enter Block: ");
       Scanf (" 1. [ " In]", block);
       add Allotment (id, name, room, block);
       break;
case 2:
      printf (" Enter the student 10 to remove : "):
      s canf ( 1. [ 1/h] ", id);
       nemove Student (id);
       break;
Case 3:
        prints ("Enter Name to search: ");
        starf (" 1. [ ] ", name);
         search By Name (name);
         break;
Case 4:
      . prints ( Enter Room Number to search ; ");
         scanf ("1.d", 2 noom);
         search Byroom (room);
```

bnoak:

```
case 5:
       prints ("Enter Block to display: ");
       scanf (" 1.E", block);
       display Blockwise (blow);
       break;
case 6:
       prints ("Reverse Display; \n");
       neverse Display (head);
        break;
 case 1:
       donedhist : done List (head);
        prints (" List cloned successfully. \n");
         break;
 case 8:
        Count PerBlock ();
         break:
 case on:
        enit (0);
default :
       printf ("Invalid Choice. Try again. \n");
   3
3
  neturn 0;
3
```

Student hostel allotment system-71052410040 -Assignment1

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
typedef struct Student {
  char studentID[20]; // now string, not int
  char name[50];
  int roomNo:
  char block[10];
  struct Student* next;
} Student;
Student* head = NULL;
// Function to create new node
Student* createNode(char id[], char name[], int room,
char block[]) {
  Student* newNode =
(Student*)malloc(sizeof(Student));
  strcpy(newNode->studentID, id);
  strcpy(newNode->name, name);
  newNode->roomNo = room;
  strcpy(newNode->block, block);
  newNode->next = NULL;
  return newNode;
}
// Add new allotment
void addAllotment(char id[], char name[], int room, char
block[]) {
  Student* newNode = createNode(id, name, room,
block);
  if (head == NULL) {
    head = newNode;
  } else {
    Student* temp = head;
    while (temp->next != NULL)
      temp = temp->next;
    temp->next = newNode;
  printf("Allotment added successfully!\n");
}
```

```
// Remove student by ID
void removeStudent(char id[]) {
  if (head == NULL) {
    printf("No records found.\n");
    return;
  Student* temp = head;
  Student* prev = NULL;
  while (temp != NULL && strcmp(temp->studentID, id) !
= 0) {
    prev = temp;
    temp = temp->next;
  if (temp == NULL) {
    printf("Student ID %s not found.\n", id);
    return;
  }
  if (prev == NULL) {
    head = head->next;
  } else {
    prev->next = temp->next;
  free(temp);
  printf("Student with ID %s removed successfully.\n",
id);
}
// Search by name
void searchByName(char name[]) {
  Student* temp = head;
  int found = 0;
  while (temp != NULL) {
    if (strcmp(temp->name, name) == 0) {
      printf("Found: ID=%s, Name=%s, Room=%d,
Block=%s\n",
          temp->studentID, temp->name,
temp->roomNo, temp->block);
      found = 1;
    }
    temp = temp->next;
  if (!found) printf("No student with name %s found.\n",
name);
}
// Search by room number
void searchByRoom(int room) {
  Student* temp = head;
  int found = 0;
  while (temp != NULL) {
    if (temp->roomNo == room) {
      printf("Found: ID=%s, Name=%s, Room=%d,
Block=%s\n",
          temp->studentID, temp->name,
temp->roomNo, temp->block);
      found = 1;
    }
    temp = temp->next;
  if (!found) printf("No student found in room %d.\n",
room);
}
```

```
// Display block-wise
void displayBlockWise(char block[]) {
  Student* temp = head;
  int found = 0;
  printf("Students in Block %s:\n", block);
  while (temp != NULL) {
    if (strcmp(temp->block, block) == 0) {
      printf("ID=%s, Name=%s, Room=%d\n",
          temp->studentID, temp->name,
temp->roomNo);
      found = 1;
    temp = temp->next;
  if (!found) printf("No students in block %s.\n", block);
}
// Reverse display using recursion
void reverseDisplay(Student* node) {
  if (node == NULL) return;
  reverseDisplay(node->next);
  printf("ID=%s, Name=%s, Room=%d, Block=%s\n",
      node->studentID, node->name, node->roomNo,
node->block);
// Clone list
Student* cloneList(Student* head) {
  if (head == NULL) return NULL;
  Student* newHead = NULL, *tail = NULL;
  Student* temp = head;
  while (temp != NULL) {
    Student* newNode = createNode(temp->studentID,
temp->name, temp->roomNo, temp->block);
    if (newHead == NULL) {
      newHead = newNode;
      tail = newHead;
    } else {
      tail->next = newNode;
      tail = newNode;
    temp = temp->next;
  return newHead;
// Count students per block
void countPerBlock() {
  Student* temp = head;
  int countA = 0, countB = 0, countC = 0, countOther =
0;
  while (temp != NULL) {
    if (strcmp(temp->block, "A") == 0) countA++;
    else if (strcmp(temp->block, "B") == 0) countB++;
    else if (strcmp(temp->block, "C") == 0) countC++;
    else countOther++;
    temp = temp->next;
  }
  printf("Block A: %d students\n", countA);
  printf("Block B: %d students\n", countB);
  printf("Block C: %d students\n", countC);
  printf("Other Blocks: %d students\n", countOther);
}
```

```
int main() {
  int choice, room;
  char id[20], name[50], block[10];
  Student* clonedList = NULL;
  while (1) {
    printf("\n--- Student Hostel Allotment System
---\n'');
    printf("1. Add New Allotment\n");
    printf("2. Remove Student\n");
    printf("3. Search by Name\n");
    printf("4. Search by Room Number\n");
    printf("5. Display Allotments Block-wise\n");
    printf("6. Reverse Display\n");
    printf("7. Clone List\n");
    printf("8. Count Students per Block\n");
    printf("9. Exit\n");
    printf("Enter your choice: ");
    scanf("%d", &choice);
    switch(choice) {
      case 1:
         printf("Enter Student ID: ");
         scanf(" %[^\n]", id);
         printf("Enter Name: ");
         scanf(" %[^\n]", name);
         printf("Enter Room Number: ");
         scanf("%d", &room);
         printf("Enter Block: ");
         scanf(" %[^\n]", block);
         addAllotment(id, name, room, block);
         break;
      case 2:
         printf("Enter Student ID to remove: ");
         scanf(" %[^\n]", id);
         removeStudent(id);
         break;
      case 3:
         printf("Enter Name to search: ");
         scanf(" %[^\n]", name);
         searchByName(name);
         break;
```

```
case 4:
       printf("Enter Room Number to search: ");
      scanf("%d", &room);
      searchByRoom(room);
      break;
    case 5:
       printf("Enter Block to display: ");
      scanf(" %[^\n]", block);
      displayBlockWise(block);
      break;
    case 6:
       printf("Reverse Display:\n");
      reverseDisplay(head);
      break;
    case 7:
       clonedList = cloneList(head);
       printf("List cloned successfully.\n");
      break;
    case 8:
      countPerBlock();
      break;
    case 9:
      exit(0);
    default:
      printf("Invalid choice. Try again.\n");
return 0;
```



main.c Output





- --- Student Hostel Allotment System ---
- 1. Add New Allotment
- 2. Remove Student
- 3. Search by Name
- 4. Search by Room Number
- 5. Display Allotments Block-wise
- Reverse Display
- 7. Clone List
- 8. Count Students per Block
- 9. Exit

Enter your choice: 1

Enter Student ID: 24BAD004

Enter Name: Arthi

Enter Room Number: 107

Enter Block: A

Allotment added successfully!

- --- Student Hostel Allotment System ---
- 1. Add New Allotment
- 2. Remove Student
- 3. Search by Name
- Search by Room Number
- 5. Display Allotments Block-wise
- 6. Reverse Display
- 7. Clone List
- 8. Count Students per Block
- 9. Exit

Enter your choice: 1

Enter Student ID: 24BAD019

Enter Name: Gabri

Enter Room Number: 207

Enter Block: A

Allotment added successfully!

- --- Student Hostel Allotment System ---
- 1. Add New Allotment
- 2. Remove Student
- Search by Name
- 4. Search by Room Number
- 5. Display Allotments Block-wise
- 6. Reverse Display
- 7. Clone List
- 8. Count Students per Block
- 9. Exit

Enter your choice: 1

Enter Student ID: 24CSE046

Enter Name: Yazhini Enter Room Number: 101

Enter Block: B

Allotment added successfully!

- --- Student Hostel Allotment System ---1. Add New Allotment

 - Remove Student
 - 3. Search by Name
 - 4. Search by Room Number
 - Display Allotments Block-wise
 - 6. Reverse Display
 - 7. Clone List
 - 8. Count Students per Block
 - 9. Exit

Enter your choice: 3

Enter Name to search: Gabri

Found: ID=24BAD019, Name=Gabri, Room=207, Block=A

- --- Student Hostel Allotment System ---
- 1. Add New Allotment
- Remove Student
- 3. Search by Name
- 4. Search by Room Number
- 5. Display Allotments Block-wise
- 6. Reverse Display
- 7. Clone List
- 8. Count Students per Block
- 9. Exit

Enter your choice: 4

Enter Room Number to search: 105

No student found in room 105.

- --- Student Hostel Allotment System ---
- 1. Add New Allotment
- 2. Remove Student
- Search by Name
- 4. Search by Room Number
- 5. Display Allotments Block-wise
- 6. Reverse Display
- Clone List
- 8. Count Students per Block
- 9. Exit

Enter your choice: 5

Enter Block to display: A

Students in Block A:

ID=24BAD004, Name=Arthi, Room=107

ID=24BAD019, Name=Gabri, Room=207