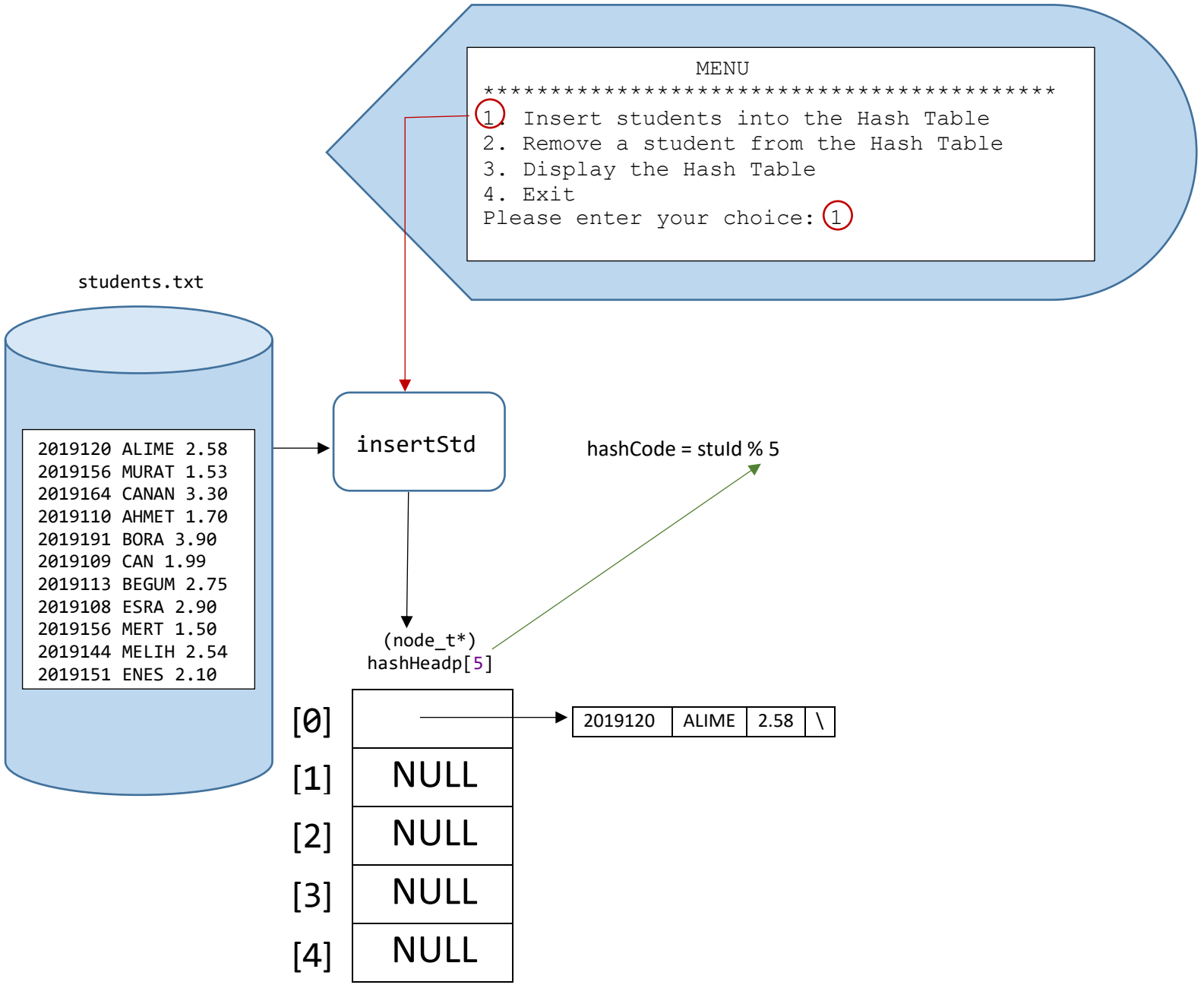
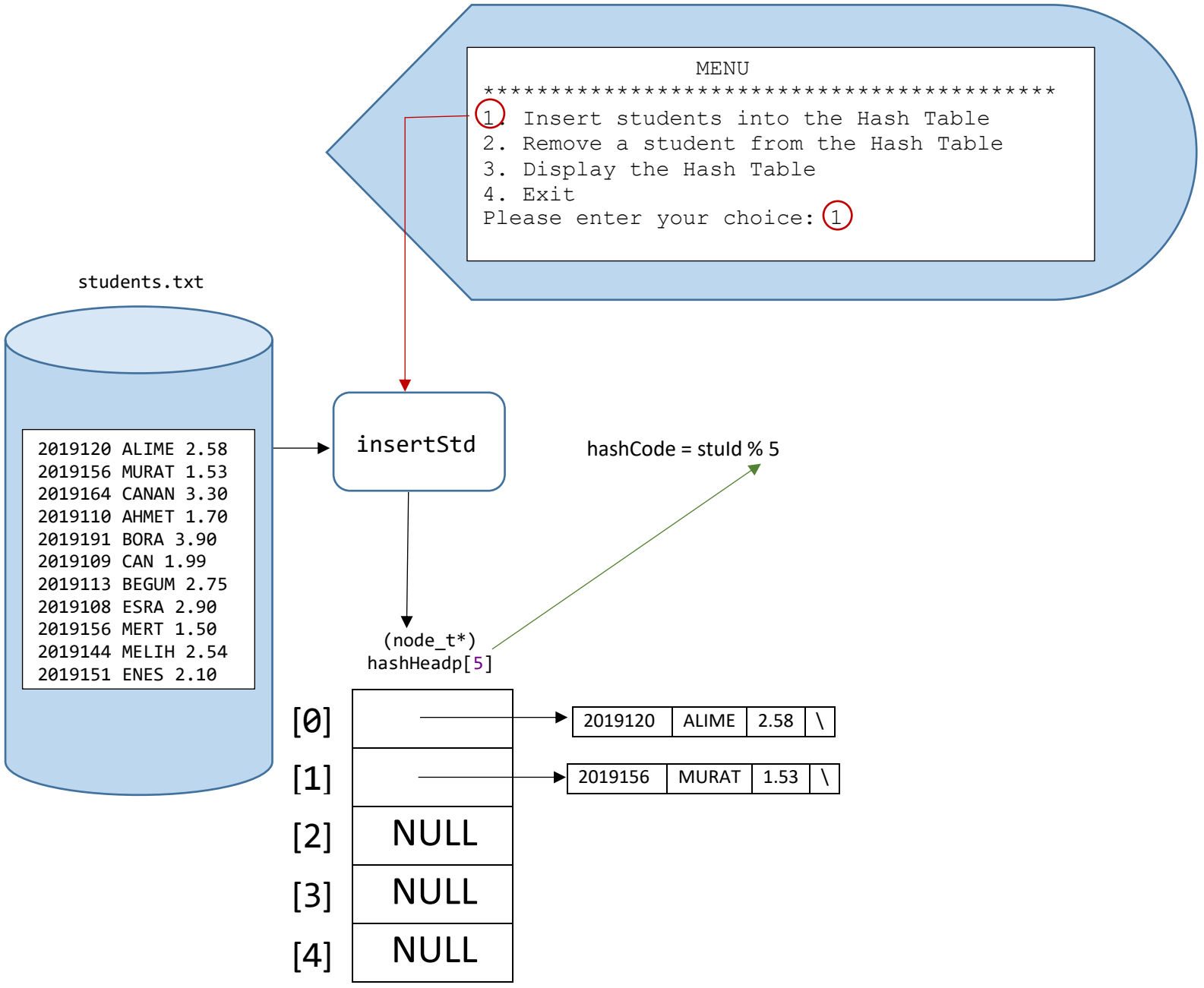
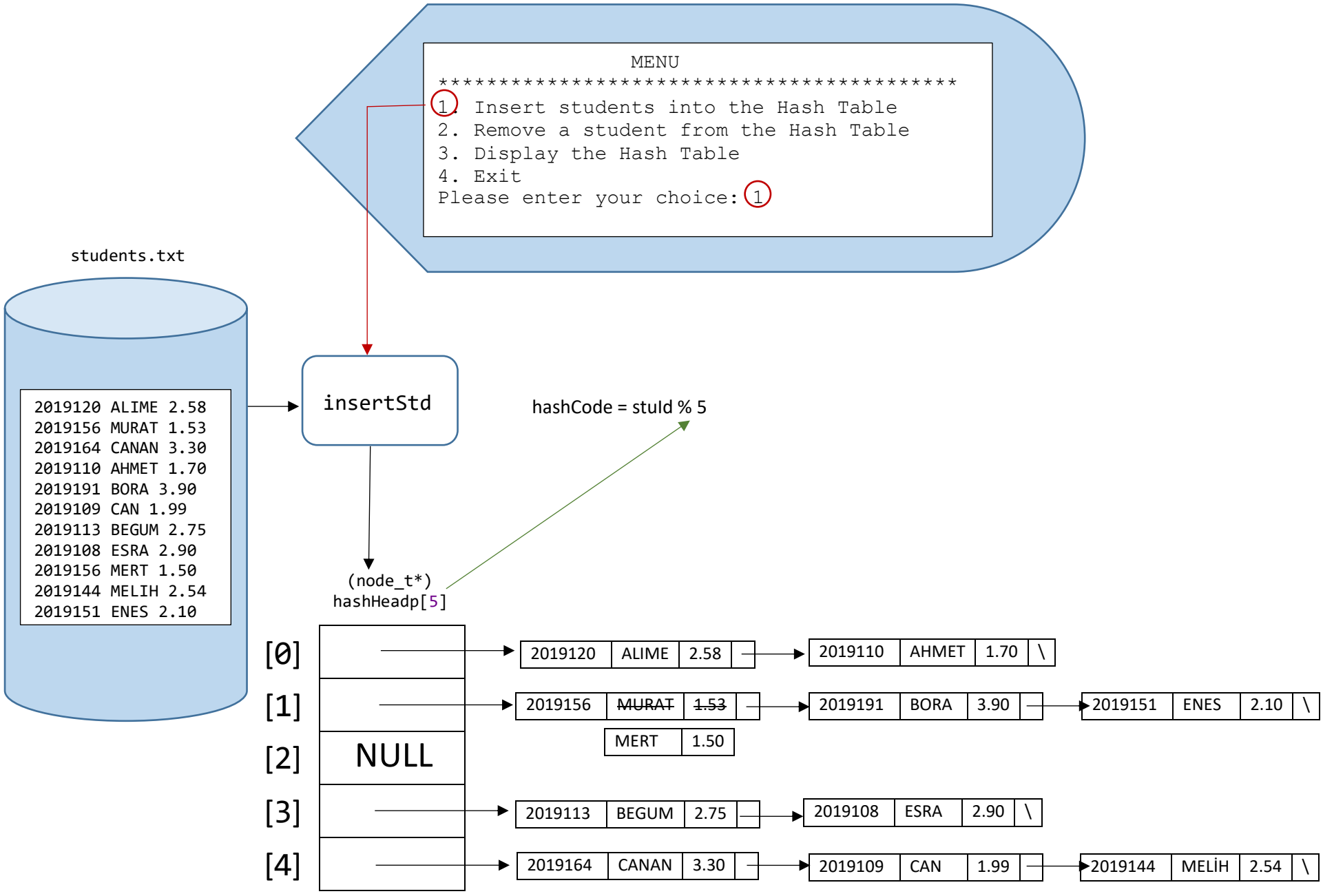
Hashed Linked List Design Example

The following structure has been used in the given example:

```
typedef struct {
	int stuId;
	char name[20];
	double cgpa;
}stu_t;
```


MENU
****
1. Insert students into the Hash Table
2. Remove a student from the Hash Table
3. Display the Hash Table
4. Exit
Please enter your choice: 1
students.txt
2019120 ALIME 2.58
2019156 MURAT 1.53
2019164 CANAN 3.30
2019110 AHMET 1.70
2019191 BORA 3.90
2019109 CAN 1.99
2019113 BEGUM 2.75
2019108 ESRA 2.90
2019156 MERT 1.50
2019144 MELIH 2.54
2019151 ENES 2.10
insertStd
hashCode = stuId % 5
(node_t*)
hashHeadp[5]
[0]
[1] NULL
[2] NULL
[3] NULL
[4] NULL
2019120
ALIME
2.58
\


MENU
****
1. Insert students into the Hash Table
2. Remove a student from the Hash Table
3. Display the Hash Table
4. Exit
Please enter your choice: 1
students.txt
2019120 ALIME 2.58
2019156 MURAT 1.53
2019164 CANAN 3.30
2019110 AHMET 1.70
2019191 BORA 3.90
2019109 CAN 1.99
2019113 BEGUM 2.75
2019108 ESRA 2.90
2019156 MERT 1.50
2019144 MELIH 2.54
2019151 ENES 2.10
insertStd
hashCode = stuId % 5
(node_t*)
hashHeadp[5]
[0]
[1]
[2]
[3]
[4]
NULL
NULL
NULL
2019120
ALIME
2.58
\
2019156
MURAT
1.53
\


MENU
****
1. Insert students into the Hash Table
2. Remove a student from the Hash Table
3. Display the Hash Table
4. Exit
Please enter your choice: 1
students.txt
2019120 ALIME 2.58
2019156 MURAT 1.53
2019164 CANAN 3.30
2019110 AHMET 1.70
2019191 BORA 3.90
2019109 CAN 1.99
2019113 BEGUM 2.75
2019108 ESRA 2.90
2019156 MERT 1.50
2019144 MELIH 2.54
2019151 ENES 2.10
insertStd
hashCode = stuId % 5
(node_t*)
hashHeadp[5]
[0]
[1]
[2]
[3]
[4]
NULL
2019120 ALIME 2.58
2019110 AHMET 1.70 \
2019156 MURAT 1.53
2019191 BORA 3.90
2019151 ENES 2.10 \
MERT 1.50
2019113 BEGUM 2.75
2019108 ESRA 2.90 \
2019164 CANAN 3.30
2019109 CAN 1.99
2019144 MELİH 2.54 \

```c
#include "linked_list.h"
#define MAX 5

int menu(void)
{
   int choice;

   printf("\n\t\tMENU\n"
      "****************************************"
      "\n1. Insert students into the Hash Table"
      "\n2. Remove a student from the Hash Table"
      "\n3. Display the Hash Table"
      "\n4. Exit\n");
   do {
      printf("Please enter your choice : ");
      scanf("%d", &choice);
   } while (choice < 1 || choice>4);
   return choice;
}


void initArray(node_t *Hp[])
{
   int i;
   for (i = 0; i < MAX; i++)
      Hp[i] = NULL;
}


int hashCode(int sId)
{
   return(sId % MAX);
}
```

```
node_t *searchStdId(node_t *p, int sId)
{
   node_t *temp = p;
   while (temp != NULL && temp->data.stuId != sId)
      temp = temp->next;
   return temp;
}



void insertStds(FILE *inp, node_t *hashHeadp[])
{
   LType item;
   while (fscanf(inp, "%d %s %lf", &item.stuId, item.name, &item.cgpa) != EOF)
   {
      int index = hashCode(item.stuId); //get the position

      if (hashHeadp[index] == NULL)
         hashHeadp[index] = addBeginning(hashHeadp[index], item);
      else
      {
         node_t *addr = searchStdId(hashHeadp[index], item.stuId);

         if (addr != NULL)
            addr->data  =item; // update the data
         else
         {
            node_t *ptr = hashHeadp[index];
            while (ptr->next != NULL)
                  ptr = ptr->next;
            addAfter(ptr, item);
         }
      }
   }
   printf("\nAll of the students in the file inserted to the Hash Table...\n");
}
```

```
void removeStu(node_t *Hp[], int sId)
{
   int index = hashCode(sId);
   LType item;

   if (Hp[index] == NULL)
      printf("This Student Id does not exists\n");
   else
   {
      if (Hp[index]->data.stuId == sId)
      {
         Hp[index] = deleteFirst(Hp[index], &item);
         printf("Student having an Id %d is removed.\n", sId);
      }
      else
      {
         node_t *ptr = Hp[index];
         while (ptr->next != NULL && ptr->next->data.stuId != sId)
            ptr = ptr->next;
         if (ptr->next != NULL)
         {
            deleteAfter(ptr, &item);
            printf("Student having an Id %d is removed.\n", sId);
         }
         else
            printf("This Student Id does not exists.\n");
      }
   }
}
```

```
void display(node_t *Hp[])
{
   int i;
   printf("\n");

   for (i = 0; i < MAX; i++)
   {
      node_t *temp = Hp[i];
      if (temp == NULL)
         printf("Head [%d] : No Elements\n", i);
      else
      {
         printf("Head [%d] : ", i);
         while (temp != NULL)
         {
            printf("(%d %-10s %6.2f) -> ", temp->data.stuId, temp->data.name, temp->data.cgpa);
            temp = temp->next;
         }
         printf("NULL\n");
      }
   }
}
```

```
int main(void)
{
   FILE *inp = fopen("student.txt", "r");

   if (inp == NULL)
      printf("The file does NOT exist!!!\n");
   else {

      int choice, stuId;
      node_t *hashHeadp[MAX];
      initArray(hashHeadp);

      do {
         choice = menu();

         switch (choice)
         {
            case 1:
                  insertStds(inp, hashHeadp);
                  break;
            case 2:
                  printf("\nEnter the Stu Id to delete : ");
                  scanf("%d", &stuId);
                  removeStu(hashHeadp, stuId);
                  break;
            case 3:
                  display(hashHeadp);
                  break;
         }
      } while (choice != 4);

   }
   return(0);
}
```