```
#include <stdio.h>
#include <iostream>
using namespace std;

struct student
{
	int cod;
	char* nume;
	float medie;
};

struct hashT
{
	student** vect;
	int size;
};

int functieHash(int cheie, hashT tabela)
{
	return cheie % tabela.size;
}


int inserare(hashT tabela, student* s)
{
	int pozitie;
	if (tabela.vect != NULL)
	{
		pozitie = functieHash((*s).cod, tabela);
		if (tabela.vect[pozitie] == NULL)
			tabela.vect[pozitie] = s;
		else
		{
			int index = 1;
			while (pozitie + index < tabela.size)
			{
				if (tabela.vect[pozitie + index] == NULL)
				{
					tabela.vect[pozitie + index] = s;
					pozitie += index;
					break;
				}
				index++;
			}
```

```
			}
		}
		return pozitie;
}

void traversare(hashT tabela)
{
		if (tabela.vect != NULL)
		{
			for (int i = 0; i < tabela.size; i++)
				if (tabela.vect[i] != NULL)
				{
					printf("\nPozitie: %d", i);
					printf("\nCod=%d, Nume=%s, Medie=%5.2f",
tabela.vect[i]->cod, tabela.vect[i]->nume, tabela.vect[i]->medie);
				}
		}
}

void dezalocare(hashT tabela)
{
		if (tabela.vect != NULL)
		{
			for (int i = 0; i < tabela.size; i++)
				if (tabela.vect[i] != NULL)
				{
					free(tabela.vect[i]->nume);
					free(tabela.vect[i]);
				}
			free(tabela.vect);
		}
}

int stergere(hashT tabela, int cod)
{
		int pozitie;
		if (tabela.vect != NULL)
		{
			pozitie = functieHash(cod, tabela);
			if (tabela.vect[pozitie] == NULL)
				return -1;
			else
				if (tabela.vect[pozitie]->cod == cod)
				{
					free(tabela.vect[pozitie]->nume);
					free(tabela.vect[pozitie]);
```

```
                    tabela.vect[pozitie] = NULL;
                }
                else
                {
                    int index = 1;
                    while (pozitie + index < tabela.size)
                    {
                        if (tabela.vect[pozitie + index]->cod ==
cod)
                        {
                            pozitie += index;
                            free(tabela.vect[pozitie]->nume);
                            free(tabela.vect[pozitie]);
                            tabela.vect[pozitie] = NULL;
                            break;
                        }
                        index++;
                    }
                }
    }
    return pozitie;
}

void main()
{
    int n;
    printf("Nr. studenti=");
    scanf("%d", &n);

    student* s;
    char buffer[20];

    hashT tabela;
    tabela.size = 101;
    tabela.vect = (student**)malloc(tabela.size * sizeof(student*));
    for (int i = 0; i < tabela.size; i++)
        tabela.vect[i] = NULL;


    for (int i = 0; i < n; i++)
    {
        s = (student*)malloc(sizeof(student));
        printf("\nCod=");
        scanf("%d", &s->cod);

        printf("\nNume=");
```

```
            scanf("%s", buffer);

            s->nume = (char*)malloc((strlen(buffer) + 1) *
sizeof(char));
            strcpy(s->nume, buffer);

            printf("\nMedie=");
            scanf("%f", &s->medie);

            inserare(tabela, s);
      }
      traversare(tabela);

      stergere(tabela, 505);

      printf("\n----dupa stergere");

      traversare(tabela);

      dezalocare(tabela);
}
```