STIVE

```
struct carte
{
	int cod;
	char* titlu;
	float pret;
};

struct nodStiva
{
	carte inf;
	nodStiva* next;
};
void push(nodStiva** varf, carte c)
{
	nodStiva* nou = (nodStiva*)malloc(sizeof(nodStiva));
	nou->inf.cod = c.cod;
	nou->inf.titlu = (char*)malloc((strlen(c.titlu) + 1) *
sizeof(char));
	strcpy(nou->inf.titlu, c.titlu);
	nou->inf.pret = c.pret;
	nou->next = NULL;
	if (*varf == NULL)
	{
		*varf = nou;
	}
	else

	{
		nou->next = *varf;
		*varf = nou;
	}
}

int pop(nodStiva** varf, carte* val)
{
	if (*varf == NULL) return -1;
		else
	{
		(*val).cod = (*varf)->inf.cod;
```

```
            (*val).titlu = (char*)malloc((strlen((*varf)->inf.titlu) +
1) * sizeof(char));
            strcpy((*val).titlu, (*varf)->inf.titlu);
            (*val).pret = (*varf)->inf.pret;
            nodStiva* temp = *varf;
            *varf = (*varf)->next;

            free(temp->inf.titlu);
            free(temp);
            return 0;
      }
}

void traversare(nodStiva* varf)
{
      nodStiva* temp = varf;
      while (temp)
      {
            printf("\nCod = %d, Titlu = %s, Pret = %5.2f", temp-
>inf.cod, temp->inf.titlu, temp->inf.pret);
            temp = temp->next;
      }
}
struct nodLista
{
      carte inf;
      nodLista* next;
                                    };

void conversieStivaListaSimpla(nodStiva** varf, nodLista** cap)
{
      carte val;
      while (pop(varf, &val) == 0)
      {
            inserareLista(cap, val);
      }
                                     }


void conversieStivaVector(nodStiva** varf, carte* vect, int* nr)
{
      carte val;
      while (pop(varf, &val) == 0)

      {
            vect[*nr] = val;
```

```
		(*nr)++;
	}
}

void traversareLista(nodLista* cap)
{
	nodLista* temp = cap;
	while (temp)
	{
		printf("\nCod = %d, Titlu = %s, Pret = %5.2f", temp->inf.cod, temp->inf.titlu, temp->inf.pret);
		temp = temp->next;
	}
}


void main()
{
	nodStiva* varf = NULL;
	int n;
	carte c;
	printf("Nr. carti=");
	scanf("%d", &n);
	char buffer[20];
	for (int i = 0; i < n; i++)
	{
		printf("\nCod=");
		scanf("%d", &c.cod);
		printf("\nTitlu=");
		cin.getline(buffer, 20, ';');
		c.titlu = (char*)malloc((strlen(buffer) + 1) * sizeof(char));
		strcpy(c.titlu, buffer);
		printf("\nPret=");
		scanf("%f", &c.pret);
		push(&varf, c);
	}
	traversare(varf);

	carte val;
	pop(&varf, &val);
	printf("\nCartea extrasa are codul %d si titlul %s", val.cod, val.titlu);
	free(val.titlu);
	printf("\n----------------\n");
	/*carte *vect = (carte*)malloc(n*sizeof(carte));
```

```
	int nr = 0;
	conversieStivaVector(&varf, vect, &nr);
	for(int i=0;i<nr;i++)
		printf("\nCod = %d, Titlu = %s, Pret = %5.2f", vect[i].cod,
vect[i].titlu, vect[i].pret);
	free(vect);*/

	nodLista* cap = NULL;
	conversieStivaListaSimpla(&varf, &cap);
	traversareLista(cap);

}
```