```
#include <stdbool.h>
#define max 50
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
// 1ªquestão

typedef int tipoID;

typedef struct {
   tipoID ID;
   char matricula[20];
   char ocorrencia [10];//perda, esquecimento, não possui, outros;
}tiporegistro;

typedef struct{
   tiporegistro v[max];
   int nElem;
}tipolista;



void startlista(tipolista* lista1);
int tamlista(tipolista* lista1);
void colocar_lista(tipolista* lista1, int pos, tiporegistro reg1);
void mostra_lista (tipolista* lista1);
int busca_lista(tipolista* lista1, tiporegistro chave);
void apagardalista(tipolista* lista1, tiporegistro chave);
void zerarlista(tipolista* lista1);

int main(){
        setlocale(LC_ALL, "Portuguese");
```

```
        tipolista lista1;
        tiporegistro registro;
        tiporegistro aa;

        int posicao;
        int contt;
        int op;
   do{
            printf("Bem vindo ao menu,meu querido usuário\n");
            printf("Digite o número correspondente a operação desejada!\n");
            printf("1- Criar Lista de ocorrências\n2- Adicionar ocorrência\n3- Buscar
ocorrência\n");
            printf("4- Mostrar Lista de ocorrências\n5- Apagar ocorrência\n6- Zerar lista de
ocorrências\n");
            printf("7- Quantidades de ocorrência registradas\n8- Sair\n");
            printf("Opcao: ");
            scanf("%d", &op);

        switch(op){
                case 1:
                        startlista(&lista1);

                                break;

                        case 2:
                                printf("Digite uma posição!\n");
                                scanf("%d", &posicao);
                                printf("Digite um numero de Identificacao\n");
                                scanf("%d", &registro.ID);
                                printf("Digite o nome do aluno\n");
                                scanf("%s", &registro.matricula);
                                printf("Digite um tipo de ocorrencia\n");
```

```
            scanf("%s", &registro.ocorrencia);
            colocar_lista(&lista1, posicao, registro);
            system("cls");
            break;

case 3:
            printf("Digite um ID para busca\n");
            scanf("%d", &aa);
            busca_lista(&lista1, aa);
            break;

case 4:
            mostra_lista(&lista1);
            break;

case 5:
            printf("Digite um ID para apagar\n");
            scanf("%d", &aa);
            apagardalista(&lista1, aa);
            break;

case 6:
            zerarlista(&lista1);
            break;

case 7:
            tamlista(&lista1);

case 8:
            system("cls");
            exit(1);
```

```
                                default:

                                                break;
                        }
            }while(op!=0);
    system("cls");
    return 0;
    }

void startlista(tipolista* lista1){
    lista1->nElem = 0;
    if (lista1->nElem == 0){
            printf("lista Criada\n");
            system("pause");
            system("cls");
            }
}

int tamlista(tipolista* lista1){

    return lista1->nElem;
}

void colocar_lista(tipolista* lista1, int pos, tiporegistro reg1){
    int x;
    if((lista1->nElem == max) || (pos < 0) || (pos > lista1->nElem)){
        printf("Não foi possivel adicionar!!\n");
        system("pause");
        system("cls");
    }else{
        for(x = lista1->nElem; x < pos; x--){
```

```
            lista1->v[x]=lista1->v[x-1];
            lista1->v[x]=reg1;
            lista1->nElem++;
            printf("adicionado com sucesso!!\n");
            system("pause");
            system("cls");
        }
    }
}

void mostra_lista(tipolista *lista){
            int i;
            printf("Lista: \n");
            for(i = 0; i < lista->nElem; i++){
                        printf("%d", lista->v[i].ID);
                        printf("\n");
            }
}

int busca_lista(tipolista* lista1, tiporegistro chave){
    int z;
    for (z=0; z < lista1->nElem; z++){
        if(lista1->v[z].ID == chave.ID){
            printf("Aqui esta: %d\n", lista1->v[z].ID);
            printf("Aqui esta: %s\n", lista1->v[z].matricula);
            printf("Aqui esta: %s\n", lista1->v[z].ocorrencia);
            system("pause");

        }
            else{
              printf("Não foi encontrado\n");
```

```
            system("pause");
            system("cls");
          }
    }
}

void apagardalista(tipolista* lista1, tiporegistro chave){
    int d, pos;
    pos=busca_lista(lista1, chave);
    if(pos == -1){
        printf("Não é possível excluir\n");
    }
    else{
        for(d=pos; d<lista1->nElem-1; d++){
            lista1->v[d]=lista1->v[d+1];
        }
        lista1->nElem--;
        printf("Excluido com sucesso\n");
    }

}

void zerarlista(tipolista* lista1){
            lista1->nElem=0;
}
```

```
#define max 50
#include <stdio.h>
#include <stdlib.h>
#include <locale.h>
//Programa para criar agenda de compromisssos
//Metodo de Fila First In First Out
//2ªquestão

typedef int t_chave;
typedef struct {
	int dia;
	int mes;
	int ano;
}t_data;

typedef struct {
	t_chave ID;
	t_data data;
	char desc_comprom[300];
}t_Comp;

typedef struct {
	t_Comp v[max];
	int inicio;
	int nElem;
}t_FILA;

/*INICIALIZAR LISTA 1
RETORNAR A QUANTIDADE DE ELEMENTOS VALIDOS 2
EXIBIR OS ELEMENTOS DA ESTRUTURA 3
INSERIR ELEMENTOS NA ESTRUTURA (NO FIM) 4
```

```
EXCLUIR ELEMENTOS DA ESTRUTURA (DO INÍCIO) 5
REINICIALIZAR A ESTRUTURA 6 */

//1
void inicializarLista (t_FILA* f){
	f->nElem = 0;
	f->inicio = 0;
}
//2
int numElementos (t_FILA* f){
	return f->nElem;
}

//3
void exibirFila (t_FILA* f){
	printf("Seus Compromissos\n:");
	int i = f -> inicio;
	int temp;
	for(temp = 0; temp < f->nElem; temp++){
		printf("%d\n", f->v->ID);
		printf("%d\n", f->v->data);
		printf("%s\n", f->v->desc_comprom	);
		i = (i+1) % max;
	}
	printf("\n\n");
}

//4
void inserirElementos (t_FILA* f, t_Comp comp){
	if (f->nElem >= max){
		printf("Não foi possível adicional compromisso\n");
```

```
	}else{
		int pos = (f->nElem + f->inicio)% max;
		f->v[pos] = comp;
		f->nElem++;
		printf("Adicionado com sucesso!!\n");
	}
}

//5
void excluirElemento (t_FILA* f, t_Comp* comp ){
	if (f->nElem==0){
		printf("A lista está vazia");
	}else{
		*comp = f->v[f->inicio];
		f->inicio = (f->inicio+1) % max;
		f->nElem--;
		printf("Excluido com sucesso!\n");
	}
}

//6
void reiniciarFila(t_FILA* f){
	inicializarLista(f);
}

int main(){
	setlocale(LC_ALL, "Portuguese");
	printf("Se apareceu aqui ta de boas lá em cima !\n");

	t_Comp Compro;
	t_FILA f;
```

```
        int op;
   do{
            printf("Bem vindo ao menu,meu querido usuário\n");
            printf("Digite o número correspondente a operação desejada!\n");
            printf("1- Criar Lista de ocorrências\n2- Adicionar compromisso \n3- Mostrar
Compromissos\n");
            printf("4- Apagar ocorrência \n5- Zerar lista de ocorrências \n6- Quantidades de
ocorrência registradas \n");
            printf("7- Sair\n");
            printf("Opcao: ");
            scanf("%d", &op);

        switch(op){
                case 1:
                        inicializarLista(&f);

                                break;

                        case 2:
                                Compro.ID = f.nElem;
                                printf("Digite uma data!\n");
                                scanf("%d", &Compro.data);
                                printf("Digite uma descrição para o compromisso \n");
                                scanf("%s", &Compro.desc_comprom);
                                inserirElementos(&f, Compro);
                                system("cls");
                                break;

                        case 4:
                                exibirFila(&f);
                                break;
```

```
                case 5:
                        printf("Digite o ID do compromisso a ser excluido!\n");
                        scanf("%d", Compro.ID);
                        excluirElemento(&f, Compro.ID);
                        break;

                case 6:
                        reiniciarFila(&f);
                        break;

                case 7:
                        numElementos(&f);

                case 8:
                        system("cls");
                        exit(1);
                default:

                        break;
        }
    }while(op!=0);


}
```