5) Considere o tipo Candidato abaixo:

```
struct candidato
{       int     inscrição;
        char    nome[51];
        int     idade;
        float   nota;
};
typedef struct candidato Candidato;
```

Escreva as funções:

1) criaCandidato: que recebe uma inscrição, um nome, uma idade e uma nota, e cria, com alocação dinâmica, um novo candidato, preenchendo seus campos. A função retorna o endereço do novo candidato criado, ou, caso não tenha sido possível criar o candidato, a função retorna NULL;

2) exibeTodosCandidatos: a função recebe um vetor de ponteiros para candidatos e o número de candidatos, exibindo os dados de todos os candidatos;

3) buscaInscricao: a função recebe um vetor de ponteiros para candidatos, o número de candidatos e uma inscrição. A função retorna o endereço do candidato, se ele for encontrado, ou NULL, caso contrário.

Use a função main abaixo para testar as funções desenvolvidas.

```
int main (void)
{
      Candidato *vpcandidato[8];
      int i;
      Candidato *pont;

      vpcandidato[0]= criaCandidato(444,"Luiz",34, 7.8f);
      vpcandidato[1]= criaCandidato(111,"Rita",56, 8.8f);
      vpcandidato[2]= criaCandidato(999,"Rute",32, 7.1f);
      vpcandidato[3]= criaCandidato(555,"Dina",27, 6.4f);
      vpcandidato[4]= criaCandidato(777,"Lana",35, 5.3f);
      vpcandidato[5]= criaCandidato(666,"Tais",29, 9.8f);
      vpcandidato[6]= criaCandidato(222,"Cris",31, 7.2f);
      vpcandidato[7]= criaCandidato(333,"Vera",44, 5.4f);

      exibeTodosCandidatos(vpcandidato, 8);

/* inclua aqui a parte referente ao teste da função
buscaInscricao: leitura de uma inscrição, chamada da função e
exibição dos dados do candidato, se encontrado    */

      for (i=0; i<8; i++)
            free(vpcandidato[i]);

      return 0;
}
```