

Respostas Teóricas

Estéfano Alfredo Pavan Gonçalves

September 15, 2025

- 1 **Apresente um programa em C que reproduz o diagrama de memória dado. Coloque todos os comandos na função main. Forneça você mesmo a struct adequada para o exemplo.**

Struct Utilizada:

```
typedef struct Tno
{
    int chave;
    struct Tno *esq;
    struct Tno *dir;
}Tno;
```

listing

```
int main()
{
    Tno *raiz = NULL;

    //No chave = 7
    /*
     * N o utilizarei cast em nenhum malloc,
     * Segundo visto em sala,
     * o tipo da variavel funciona de
     * forma semelhante para esse caso.
     */
    raiz = malloc(sizeof(Tno));
    if(raiz)
    {
        raiz->chave = 7;
        raiz->esq = NULL;
        raiz->dir = NULL;
    }
    else return -1;
```

```

//No chave = 5.
raiz->esq = malloc(sizeof(Tno));
if(raiz->esq)
{
    raiz->esq->chave = 5;
    raiz->esq->esq = NULL;
    raiz->esq->dir = NULL;
}
else return -1;

//No chave = 2.
raiz->esq->esq = malloc(sizeof(Tno));
if(raiz->esq->esq)
{
    raiz->esq->esq->chave = 2;
    raiz->esq->esq->esq = NULL;
    raiz->esq->esq->dir = NULL;
}
else return -1;

//No chave = 6.
raiz->esq->dir = malloc(sizeof(Tno));
if(raiz->esq->dir)
{
    raiz->esq->dir->chave = 6;
    raiz->esq->dir->esq = NULL;
    raiz->esq->dir->dir = NULL;
}
else return -1;

//No chave = 9.
raiz->dir = malloc(sizeof(Tno));
if(raiz->dir)
{
    raiz->dir->chave = 9;
    raiz->dir->esq = NULL;
    raiz->dir->dir = NULL;
}
else return -1;

//No chave = 8.
raiz->dir->esq = malloc(sizeof(Tno));
if(raiz->dir->esq)
{
    raiz->dir->esq->chave = 8;
    raiz->dir->esq->esq = NULL;
    raiz->dir->esq->dir = NULL;
}
else return -1;

```

```
}  
    return 0;
```

- 2** Apresente o comando para imprimir na tela o endereço de memória do nó raiz 7 e o endereço da variável "raiz". Em ambos os casos, utilize a variável "raiz" para imprimir os valores requisitados.

```
//imprime o endereço do nó raiz 7, através de raiz.  
printf("%p", raiz);  
  
//imprime o endereço de raiz, através de raiz.  
printf("%p", &raiz);
```

- 3** Apresente o comando para imprimir na tela o endereço de memória do nó raiz 7 e o endereço da variável "raiz". Em ambos os casos, utilize a variável "r" para imprimir os valores requisitados.

```
//imprime endereço do nó raiz 7  
printf("%p", *r);  
  
//imprime na tela o endereço da variável raiz  
printf("%d", r);
```

- 4** Apresente o comando para imprimir na tela o valor-chave do nó raiz 7 duas vezes. Em uma impressão, utilize a variável "raiz". Na outra, use a variável "r".

```
printf("%d %d", raiz->chave, (*r)->chave);
```

- 5 Apresente o comando para imprimir na tela o valor do campo `esq` do nó 7 e o endereço onde reside o campo `dir` do mesmo nó. Em ambos os casos, utilize a variável `"raiz"` para imprimir os valores requisitados.

```
printf("%p %p", raiz->esq, &(raiz->dir));
```

- 6 Apresente o comando para imprimir na tela o valor do campo `esq` do nó 7 e o endereço onde reside o campo `dir` do mesmo nó. Em ambos os casos, utilize a variável `"r"` para imprimir os valores requisitados.

```
printf("%p %p", (*r)->esq, &((*r)->dir));
```

- 7 Apresente o comando para remover corretamente o nó 8 da árvore. Utilize a variável `"raiz"` para realizar a remoção.

```
free(raiz->dir->esq);  
raiz->dir->esq = NULL;
```

- 8 Apresente o comando para remover corretamente o nó 8 da árvore. Utilize a variável `"r"` para realizar a remoção.

```
free((*r)->dir->esq);  
(*r)->dir->esq = NULL;
```