**1) Crie um registro célula contendo os atributos elemento (inteiro) e prox (apontador para outra célula):**

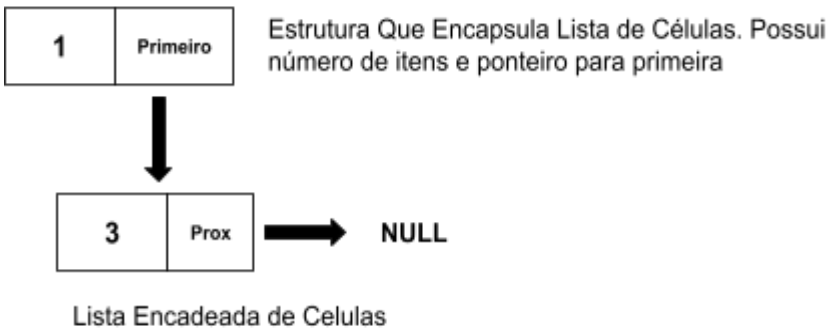
**R:**

```
typedef int dado_t;

typedef struct Celula_s {
        dado_t dado;
        struct Celula_s* prox;
}Celula_t;

typedef struct ListaCelula_s {
        uint32_t numCelulas;
        Celula_t *primeiro;
} ListaCelula_t;

void insere(ListaCelula_t *celulas, dado_t valor){
        Celula_t *novo = malloc(sizeof(Celula_t));
        if(novo != NULL){
                novo->dado = valor;
                novo->prox = celulas->primeiro;

                celulas->numCelulas++;
                celulas->primeiro = novo;
        }
}
```

**2) Mostre o que acontece se um método tiver o comando Celula *tmp = novaCelula(3).**

**R:**

```
        insere(&celulas, 3);
```



**3) Represente graficamente o código Java abaixo**

```
Elemento e1;
```

**R:**


**4) Represente graficamente o código Java abaixo**

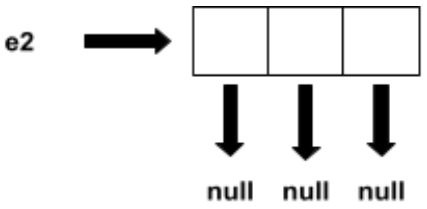
```
Elemento e1 = new Elemento();
```

**R:**


**5) Represente graficamente o código Java abaixo**

```
Elemento[ ] e2 = new Elemento[3];
```

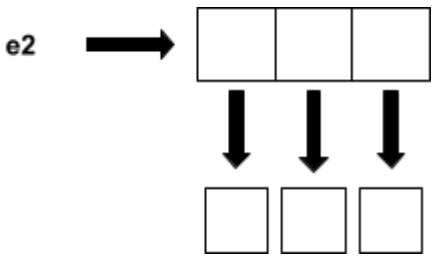
**R:**




**6) Represente graficamente o código Java abaixo**

```
Elemento[] e2 = new Elemento[3];
for (int i = 0; i < 3; i++) {
        e2[i] = new Elemento();
}
```

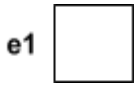
**R:**



**7) Represente graficamente o código C abaixo**

Elemento e1;

**R:**




**8) Represente graficamente o código C abaixo**

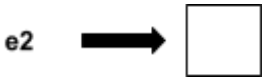
Elemento* e2;

**R:**


**9) Represente graficamente o código C abaixo**

Elemento* e2 = (Elemento*) malloc(sizeof(Elemento));

**R:**




**10) Represente graficamente o código C abaixo**

Elemento* e2 = (Elemento*) malloc(3*sizeof(Elemento));

**R:**


**11) Represente graficamente o código C abaixo**

Elemento e3[3];

**R:**

**12) Represente graficamente o código C abaixo**

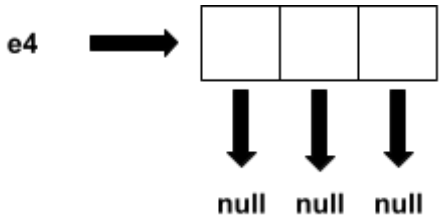
```
Elemento** e4;
```

**R:**


**13) Represente graficamente o código C abaixo**

```
Elemento** e4 = (Elemento**) malloc(3*sizeof(Elemento*));
```

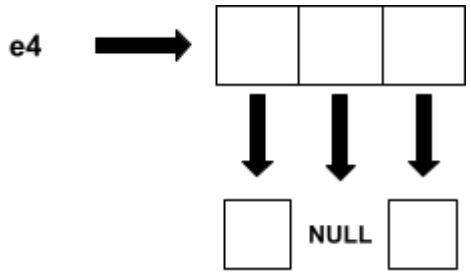
**R:**




**14) Represente graficamente o código C abaixo**

```
Elemento** e4 = (Elemento**) malloc(3*sizeof(Elemento*));

e4[0] = (Elemento*) malloc(sizeof(Elemento*));
e4[2] = (Elemento*) malloc(sizeof(Elemento*));
```

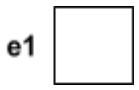
**R:**




**15) Represente graficamente o código C++ abaixo**

```
Elemento e1;
```

**R:**



**16) Represente graficamente o código C++ abaixo**

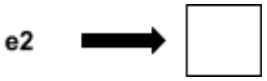
```
Elemento* e2;
```

**R:**


**17) Represente graficamente o código C++ abaixo**

```
Elemento* e2 = new Elemento;
```

**R:**




**18) Represente graficamente o código C++ abaixo**

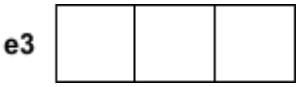
```
Elemento* e2 = new Elemento[3];
```

**R:**


**19) Represente graficamente o código C++ abaixo**

```
Elemento e3[3];
```

**R:**




**20) Represente graficamente o código C++ abaixo**

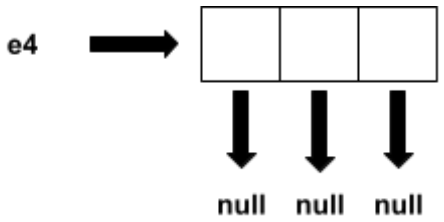
```
Elemento** e4;
```

**R:**

**21) Represente graficamente o código C++ abaixo**

```
Elemento** e4 = new Elemento*[3];
```

**R:**




**22) Represente graficamente o código C++ abaixo**

```
Elemento** e4 = new Elemento*[3];

e4[0] = new Elemento;
e4[2] = new Elemento;
```

**R:**