

## Exercícios 28-03

1) Identificar os problemas dos códigos abaixo:

a)

```
main() {  
    int v[10];  
    free(v);  
}
```

b)

```
main() {  
    int x, *p;  
    p = malloc(sizeof(int));  
    p = &x;  
    free(p);  
}
```

c)

```
main(){  
    int *v;  
    v = malloc (10 * sizeof (int));  
    free(v+1);  
}
```

2) Verifique o que faz o código abaixo. Ele executa corretamente?

```
main() {  
    char *s, a[] = "Algoritmos";  
    int t;  
    s = malloc(30 * sizeof(int));  
    strcpy(s, a);  
    t = strlen(s);  
    printf("Tamanho: %d\n", t);  
    s = realloc(s, t * sizeof(int));  
    free(s);  
}
```

E se a seguinte alteração for feita, o que acontece?

```

main() {
    char *s, a[] = "Algoritmos";
    int t;
    s = malloc(30 * sizeof(int));
    s = a;
    t = strlen(s);
    printf("Tamanho: %d\n", t);
    s = realloc(s, t * sizeof(int));
    free(s);
}

```

**3)** Crie um vetor do tipo inteiro com 50 itens usando a função malloc. Percorra o vetor imprimindo endereço e valor de cada item.

**4)** Repetir o exercício 2 usando a função calloc em vez de malloc.

**5)** Criar um TAD para representar frações. O TAD deve conter uma estrutura como a do exemplo abaixo:

```

struct fracao {
    int numerador;
    int denominador;
};

```

e funções para:

- Criar uma fração;
- Liberar uma fração;
- Somar duas frações;
- Subtrair duas frações;
- Multiplicar duas frações;
- Dividir duas frações;
- Imprimir fração;