Lista 0
Revisão de comandos básicos de C

1. Saída de dados com `printf`.

```
int main(){
    char x,y;
    int z;
    float w;
    x = 65;
    y = 'B';
    z = 345;
    w = 4.567;
    printf("Apresentacao de caracteres 1: %c, %c \n", x, y);
    printf("Apresentacao de caracteres 2: %d, %d \n", x, y);
    printf("Inteiro: %d\nReal: %.3f", z, w);
    return 0;
}
```

2. Incremento e decremento.

```
int main(){
    char x,y;
    x = 65;
    y = 'B';
    printf("Valor original: %c (%d), %c (%d)\n", x,x,y,y);
    printf("Apresentacao com incremento posterior: %c, %c\n", x++,y--);
    printf("Apresentacao com incremento anterior: %c, %c\n", ++x, --y);
    return 0;
}
```

3. Ponteiros
   Exemplo 1

```
int main(){
    int *a;
    a = malloc(sizeof(int));
    *a = 3;
    printf("Valor do ponteiro: %d\n", a);
    printf("Valor local do apontador: %d \n", *a);
    free(a);
    return 0;
}
```

   Exemplo 2

```
int main(){
    int a, *b;
    a = 3;
    b = &a;
    *b += 5;
    printf("Valor do a: %d \n", a);
    printf("Valor do b: %d \n", *b);
    return 0;
}
```

4. Entrada de dados com `scanf`.

```
int main(){
    int v[5], i;
    for (i = 0; i < 5; i++){
        printf("Insira o %do valor:",i + 1);
        scanf("%d", &v[i]);
    }
        for(i = 0; i < 5; i++){
            printf("%d \t", v[i]);
        }
    }
```

5. Refaça o programa anterior para que a entrada de dados seja feita no formato de ponteiros, ou seja, os vetores são acessados e mostrados através de ponteiros.

6. Funções

```
void troca (int a, int b){
    int x;
    x = a;
    a = b;
    b = x;
}

int main(){
    int a, b;
    a = 3;
    b = 5;
    troca(a,b);
    printf("Valor do a: %d \n", a);
    printf("Valor do b: %d \n", b);
    return 0;
}
```

Verifique se a troca realmente ocorre.

7. No programa anterior os valores de a e b não são trocados na função *main* produzindo a saída 3 e 5. Altere o programa para que isso ocorra, usando passagem por referência e sem retorno da função. A saída do programa deverá ser 5 e 3.

8. Crie um tipo estruturado que representa um aluno, com os campos matrícula e nome completo, ambos representados por um vetor de caracteres, e uma nota, representado por um tipo float. Redefina o nome do tipo para `Aluno`.

```
struct aluno{
    char matricula[8];
    char nome[100];
    float nota;
};
typedef struct aluno Aluno;
```

   Crie um vetor do tipo `Aluno` de tamanho 3 e armazene valores no vetor. Ao final mostre os valores armazenados, sendo que cada registro seja mostrado em uma única linha, separados por um hífen.