Programação Imperativa
Teste, 23 de Maio de 2025

1. Um conjunto de inteiros pode ser representado como uma união de intervalos. Por exemplo, o conjunto {1,2,3,4,5,6,10,11} pode ser representado pela união dos intervalos [1..6] e [10..11], ou pela união dos intervalos [2..4], [1..5], [10..11] e [4..6]. Implemente a função `void ordena(Intervalo c[], int N)` que dado um conjunto `c` com `N` intervalos guardados num array, ordena os respectivos intervalos pelo limite inferior. O tipo `Intervalo` está declarado da seguinte forma.

```
typedef struct {
  int inf,sup;
```

2. Implemente a função `int cardinalidade(Intervalo c[], int N)`, que dado um conjunto `c` com `N` intervalos guardados num array e já ordenados de forma crescente pelo limite inferior, devolve quantos inteiros estão contidos nesse conjunto. No caso do conjunto ilustrado na questão anterior, a função deve devolver 8.

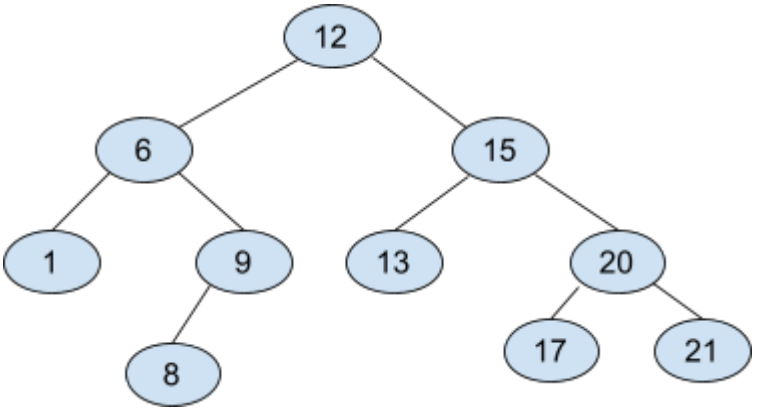
3. As duas convenções mais típicas para dar nomes a funções são conhecidas por *camel case* e *snake case*. No *camel case* as várias palavras de um identificador aparecem juntas, sendo que a primeira de cada palavra (exceto na primeira) é escrita em maiúscula. No *snake case* os identificadores são escritos sempre em minúsculas sendo as diferentes palavras separadas pelo caracter `'_'`. Por exemplo, o identificador `"apagaTodosMenores"` está escrito na convenção *camel case* e o identificador `"apaga_todos_menores"` em *snake case*. Implemente a função `void camel2snake(char *id)`, que dada uma string com um identificador na convenção *camel case* o converte para convenção *snake case*. Assuma que a string tem espaço para o identificador resultante. Note que para converter um caracter de maiúscula para minúscula basta somar-lhe 32.

4. Defina a função `LInt arrayToList (int v[], int N)` que constroi uma lista com os elementos de um array, pela mesma ordem em que aparecem no array. O tipo `LInt` está declarado da seguinte forma.

```
typedef struct no {
  int valor;
  struct no *prox;
} *LInt;
```

5. Implemente a função `int apagaUltimo(LInt *l, int x)` que dada uma lista não ordenada apaga a última ocorrência do número `x`. Se o número não existir na lista deve devolver um código de erro.

6. Implemente a função `int parentesco(ABin a, int x, int y)` que dada uma árvore binária de procura (sem números repetidos) e dois números que pertencem à árvore calcula o grau de parentesco entre eles (a distância entre eles na árvore). Por exemplo, na árvore ilustrada abaixo, o grau de parentesco entre 1 e 8 é 3. O tipo `ABin` está declarado da seguinte forma.




```
typedef struct nodo {
  int valor;
  struct nodo *esq, *dir;
} *ABin;
```