UNIVALI – EMCT KOBRASOL – Ciência da Computação – Algoritmos e Programação 2p
OUTROS EXERCÍCIOS RECURSIVIDADE – fazer em C++

RESOLUCOES

3. Verifique o que as funções abaixo mostram e retornam, e comente...

a: vai escrevendo n da vez e empilhando ateh atingir 0, qdo escreve fim e volta desempilhando

```
void func1(int n){
    if (n == 0) cout << "fim";
    else{
        cout << n;
        func1(n-1);
    }
}
```

b: vai empilhando ate atingir 0, qdo escreve fim e volta desempilhando e escrevendo o n da vez

```
void func2(int n){
    if (n == 0) cout << "fim";
    else{
        func2(n-1);
        cout << n;
    }
}
```

c: vai escrevendo n da vez e empilhando ateh atingir 0, qdo escreve fim e desce escrevendo n da vez novamente

```
void func3(int n){
    if (n == 0) cout << "fim";
    else{
        cout << n;
        func3(n-1);
        cout << n;
    }
}
```

d: vai empilhando ateh atingir 0, qdo escreve fim e desce escrevendo n da vez e empilhando novamente (arvore)

```
void func4(int n){
    if (n == 0) cout << "fim";
    else{
        func4(n-1);
        cout << n;
        func4(n-1);
    }
}
```

4. Implemente uma função recursiva que, dados dois números inteiros x e n, calcula o valor de x^n.
   Caso base? $x^0 = 1$ Passo da recursão: $x^n = x * x^{n-1}$

```
int potRec(int b, int p){
   if ( p == 0 ) return 1;
   return ( b * potRec(b, p-1));
}
```

5. Pode-se calcular o resto da divisão, MOD, de x por y, dois números inteiros positivos, usando-se a seguinte definição:
   - MOD(x,y) = MOD(x - y, y) se x > y
   - MOD(x,y) = x se x < y
   - MOD(x,y) = 0 se x = y

Caso base? São dois: x < y ou x = y Passo da recursão: MOD(x - y, y) se x > y

```
int MODRec(int x, int y){
    if(x>y)
        return MODRec(x-y,y);
    if(x<y)
        return x;
    // x==y
        return 0;
}
```

6. Usando recursividade, calcule a soma de todos os valores de um vetor de reais.
   Caso base? Tamanho do vetor = 0. Soma é 0.
   Passo da recursão: v[n-1] + soma do restante do vetor

```
float soma_v(float v[], int n){
    if(n==0) return 0;
    return v[n-1] + soma_v(v, n-1);
}
```

7. Dado um vetor de inteiros e o seu número de elementos, inverta a posição dos seus elementos.
   Caso base? Tamanho do vetor menor ou igual a 1
   Passo da recursão: troca 1o. e último elementos e inverte resto do vetor.

```
void inverteRec(int v[], int esq, int dir){
    if(esq<dir){
        swap(v[esq], v[dir]);
        inverteRec(v, esq+1, dir-1);
    }
}
```

8. Escreva uma função recursiva que determine quantas vezes um dígito K ocorre em um número natural N. P.ex., o dígito 2 ocorre 3 vezes em 762021192.
   Caso base? Quando todos os dígitos já foram examinados, ou seja, N = 0
   Passo da recursão: $n_4n_3n_2n_1n_0$ => (0 ou 1) + número de ocorrências em N / 10 ($n_4n_3n_2n_1$)

```
int conta_dig(int n, int k){
    if(n==0) return 0;
    return conta_dig(n/10, k) + (n%10==k);
}
```

9. Um problema típico em ciência da computação consiste em converter um número da sua forma decimal para a forma binária.
   Caso base? Quando o número já foi todo transformado em binário. Ou seja: x = 0
   Passo da recursão: Saber como x/2 é convertido. Depois, adicionar um dígito (0 ou 1) relativo a x.

```
void print_bin(int x){
  if(x==0) return;
  print_bin(x/2);
  cout<< x % 2;
}
// a outra é melhor
```

```
void print_bin2(int x){
  if(x==0{
     cout<<"0"; return;
  }
  print_bin(x/2);
  cout<< x%2;
}
```

10. Desenvolva subrotina recursiva para o gerador da sequência:
    F(1) = 1
    F(2) = 2
    F(n) = 2 * F(n - 1) + 3 * F(n - 2)

```
int f(int n){
  if(n==1) return 1;
  if(n==2) return 2;
  return 2*f(n-1) + 3*f(n*-2);
}
```