4) Considere um vetor com 30 números inteiros gerados aleatoriamente de 1 até 30. Faça um algoritmo para gerar este vetor e depois ordená-lo de maneira **não-crescente**.

Crie e utilize três **funções**: uma para preencher o vetor, outra para ordená-lo e uma terceira para imprimir o vetor antes e depois da ordenação.

Obs.: Não é permitido utilizar qualquer estrutura de dados para auxiliar a ordenação.

5) Considere dois números inteiros a ($a \neq 0$) e b ($b \geq 0$) lidos pelo teclado. Faça um algoritmo **recursivo** para calcular o valor de a^b.

Crie e utilize uma **função recursiva** para calcular a^b. Esta informação tem que ser impressa na função *main*.

Obs.: Não é permitido utilizar qualquer estrutura de repetição na função recursiva.

6) Considere um vetor com 20 números naturais maiores do que 1 lidos pelo teclado. Faça um algoritmo **recursivo** que organize este vetor de modo que os números **compostos** fiquem nas **primeiras** posições e os números que **não são compostos** nas **últimas** posições.

Crie e utilize duas **funções**: uma para preencher o vetor e outra recursivo para realizar a organização do mesmo. Crie e utilize também outra **função** para retornar 1, se um número natural for composto, ou retornar 0, caso contrário.

Obs. 1: Um número natural C é composto se ele tem mais de dois divisores naturais distintos;

Obs. 2: Não é permitido utilizar qualquer estrutura de dados para auxiliar a organização;

Obs. 3: Não é permitido utilizar qualquer estrutura de repetição na função recursiva.

7) Considere um vetor com 50 números inteiros gerados aleatoriamente de 1 até 100. Faça um algoritmo **recursivo** para imprimir o **maior valor** deste vetor.

Crie e utilize uma **função** para preencher o vetor e uma **função recursiva** para encontrar o maior valor do vetor. Esta informação tem que ser impressa na função *main*.

Obs. 1: Não é permitido utilizar qualquer estrutura de dados auxiliar;

Obs. 2: Não é permitido utilizar qualquer estrutura de repetição na função recursiva.

IMPORTANTE

1) Esta atividade deve ser feita **individualmente**;

2) **TODOS OS EXERCÍCIOS TÊM QUE SER FEITOS NO PAPEL**;

3) Cada aluno(a) deve enviar a imagem (de boa qualidade) destes exercícios até às **11h59** do dia **16/06/2023** para o e-mail:

philippeleal@yahoo.com.br

4) Após a hora e a data marcada para o envio da resposta, **NÃO É MAIS PERMITIDO ENVIÁ-LA**;

5) O e-mail considerado para correção será o **ÚLTIMO** enviado pelo(a) aluno(a) **dentro do prazo determinado**;

6) Ao enviar o e-mail, coloque como **Assunto** e **Nome do Arquivo**:

PAA-Encontro01-B-SeuNome

7) E-mails com o Assunto fora do padrão **NÃO SERÃO ACEITOS**.

Questão 04.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define N 30

void geraVetorAleatorio(int vetor[N]){
    srand(time(NULL));
    for(int i=0; i<N; i++) {
        vetor[i] = (rand() % 30) + 1;
    }
}

void ordenaVetor(int vetor[N]){

    for(int i=0; i<N; i++){
        for(int j=i; j<N; j++){
            if(vetor[j] > vetor[i]){

                int aux = vetor[i];
                vetor[i] = vetor[j];
                vetor[j] = aux;
            }
        }
    }
}

void exibeVetor(int vetor[N]){

    printf("\n\n---------------------------------------------------------\n\n");
    for(int i=0; i<N; i++){
        printf("|%02d|", vetor[i]);
    }
    printf("\n\n---------------------------------------------------------\n\n");
}

int main(){

    int vetor[N];
    geraVetorAleatorio(vetor);
    exibeVetor(vetor);
    ordenaVetor(vetor);
    exibeVetor(vetor);
    return 0;
}
```

Questão 05.c

```c
#include <stdio.h>
#include <stdlib.h>

int calculaPotencia(int a, int b){

    if(b == 0)
    return 1;
    else
    return a * calculaPotencia(a, b-1);
}

int main(){

    int a, b, resposta;
    printf("\n\n Digite o valor de (a), sendo (a=!0): ");
    scanf("%d", &a);
    printf("\n\n Digite o valor de (b), sendo (b>=0): ");
    scanf("%d", &b);
    resposta = calculaPotencia(a, b); // ((a)^b)*1;
    printf("\n\n Resultado: %02d \n\n", resposta);
    return 0;
}
```

Questão 06.c

```c
#include <stdio.h>
#include <stdlib.h>
#define N 20

void preencheVetor(int vetor[N]) {
  for(int i=0; i<N; i++) {
    printf("\n\n-------------------------------------------------------------------\n\n");
    printf("Digite a %d posicao do vetor: ", i);
    scanf("%d", &vetor[i]);
  }
}

int composto(int n){

  int i, cont = 0;
  for(i=1; i <= n; i++){
    if((n % i) == 0)
      cont++;
  }

  if(cont > 2)
  return 1;

  else
  return 0;
}

void organizaVetor(int vetor[N], int inicio, int fim) {

  if(inicio < fim){

    if(composto(vetor[inicio]) == 1){

      organizaVetor(vetor, inicio+1, fim);

    } else{

      int aux = vetor[inicio];
      vetor[inicio] = vetor[fim];
      vetor[fim] = aux;
      organizaVetor(vetor, inicio, fim-1);
    }
  }
}


int main() {

  int vetor[N];
  preencheVetor(vetor);
  organizaVetor(vetor, 0, N-1);
  printf("\n\n-------------------------Vetor Organizado-------------------------\n\n");
  for(int i=0; i<N; i++) {
    printf("|%02d|", vetor[i]);
  }
  printf("\n\n-------------------------------------------------------------------\n\n");
  return 0;
}
```

Questão 07.c

```c
#include <stdio.h>
#include <stdlib.h>
#include <time.h>
#define N 50

void preencheVetor(int vetor[N]) {
  srand(time(NULL));
  for(int i=0; i < N; i++) {
    vetor[i] = (rand() % 100) + 1;
    printf("|%d|", vetor[i]);
  }
}

int encontraMaiorValor(int vetor[N], int maior, int atual) {
  if(atual < N) {

    if(vetor[atual] > maior) {
      maior = vetor[atual];
    }
    encontraMaiorValor(vetor, maior, atual+1);

  } else {
    return maior;
  }
}

int main() {

  int vetor[N], maior = 0, atual = 0;
  preencheVetor(vetor);
  printf("\n");
  int resultado = encontraMaiorValor(vetor, maior, atual);
  printf("\n\n O maior numero e: %d \n\n", resultado);

  return 0;
}
```