Lista 2 - ICC

1. Escreva um programa que imprima o quadrado de todos os inteiros de 1 a 20. **Não** utilize a função pow() da biblioteca de matemática do C.

```
#include<stdio.h>

int
main(void)
{
    for (i = 1; i <= 20; i++)
    {
        int quad = i * i;
        printf("%d\n", quad);
    }
}
```

2. Escreva um programa para testar se a entrada proposta por um usuário é um número par ou um número ímpar. A saída deve ser escrita no terminal como "Esse número é par" ou "Esse número é ímpar".

```
#include<stdio.h>
#include<cc50.h>

int
main(void)
{
    printf("Digite um número inteiro: );
    int num = GetInt();

    if (num % 2 == 0)
        printf("Esse número é par!\n");
    else
        printf("Esse número é ímpar!\n");
}
```

3. Escreva um programa que pede dois valores inteiros ao usuário. Teste se o primeiro valor digitado é divisível pelo segundo e escreva na tela uma mensagem apropriada. Considere a verificação da divisão por zero, e apresente uma mensagem apropriada.

```
#include<stdio.h>
#include<cc50.h>

int
main(void)
{
    printf("Digite um valor inteiro: ");
    int valor1 = GetInt();

    printf("Digite outro valor inteiro: ");
    int valor2 = GetInt();

    if (valor2 == 0)
        printf(\nNão é possível realizar uma divisão por zero!\n");

    else if (valor1 % valor2 == 0)
        printf("\n%d é divisível por %d\n", valor1, valor2);

    else
        printf(\n%d não é divisível por %d\n", valor1, valor2);
}
```

4. Números primos podem ser gerados por um algoritmo conhecido como “Crivo de Eratóstenes”. Pesquise sobre esse algoritmo e o utilize na construção de um programa que gere todos os números primos até 150.

```
#include<stdio.h>

#define limite 150

int
main(void)
{
    int crivo[limite + 1]

    //Todos os números são de início primos
    for (int i = 2; i<= limite; i++)
        crivo[i] = 1;

    //Aplicação do Crivo
    for (int p = 2; p * p <= limite; p++)
    {
        if (crivo[p] == 0)
            continue;

        for (int i = p * p; i <= limite; i += p)
            crivo[i] = 0;
    }

    printf(“Os números primos até %d são: \n”. limite);

    for (int i = 2; i <= limite; i++)
    {
        if (crivo[i] == 1)
            printf(“%d “, i);
    }
    printf(“\n”);
}
```

5. Escreva um programa que calcula a média e o desvio padrão de um array que contém 15 float numbers.

```
#include<stdio.h>
#include<math.h>

#define tam 15

int
main(void)
{
    float numbers[tam];
    int i;

    printf("Digite os 15 números:\n");

    for (i = 0; i < tam; i++)
    {
        printf("Número %d\n", i + 1);
        scanf("%f", &numbers[i]);
    }

    //Cálculo média
    float soma;
    for (i = 0; i < tam; i++)
        soma += numbers[i];

    float media;
    media = soma / tam;

    //Cálculo desvio padrão
    float somadesvio;
    for (i = 0; i < tam; i++)
        somadesvio += pow(numbers[i] – media, 2);

    float desviopadrao;
    desviopadrao = sqrt(somadesvio / tam);

    printf("A média dos números digitados é %.2f\n", media);
    printf("O desvio padrão dos números digitados é %.2f\n", desviopadrao);
}
```

6. Escreva um programa que peça 10 inteiros ao usuário, armazene esses valores em um array e utilize uma função **min** que você criará para encontrar o menor desses valores.

```
#include<stdio.h>

int min (int array[], int tam)
{
    int menor = array[0]

    for (int I = 1; I < tam; i++)
    {
        if (array[i] < menor)
        {
            menor = array[i];
        }
    }
    return menor;
}

int
main(void)
{
    int numbers[10];

    printf(“Digite 10 números inteiros em seguida:\n”);
    scanf(“%d %d %d %d %d %d %d %d %d %d”, &numbers[0], &numbers[1], &numbers[2], &numbers[3], &numbers[4], &numbers[5], &numbers[6], &numbers[7], &numbers[8], &numbers[9]);

    int menor = min (numbers, 10);

    printf(“O menor valor é %d\n”, menor);
}
```

7. Escreva um programa para transpor uma matriz de entrada 2x3.

```
#include<stdio.h>

int
main(void)
{
	int matriz[2][3];
	int trans[3][2];
	int i, j;

	printf(“Digite os elementos da matriz 2x3:\n”);

	for (i = 0; i < 2; i++)
	{
		for (j = 0; j < 3; j++) {
			scanf(“%d”, &matriz[i][j]);
		}
	}

	printf(“\nMatriz Original\n”);

	for (i = 0; i < 2; i++)
	{
		for (j = 0; j < 3; j++) {
			scanf(“%d”, &matriz[i][j]);
		}
		printf(“\n”);
	}

	printf(“\nMatriz Transposta\n”);

	for (i = 0; i < 3; i++)
	{
		for (j = 0; j < 2; j++) {
			trans[i][j] = matriz[j][i];
			printf(“%d ”, trans[i][j]);
		}
		printf(“\n”);
	}
}
```

Lista 3 – ICC

1. Faça um programa que solicite um número inteiro de até 4 dígitos ao usuário e inverta a ordem de seus algarismos. Por exemplo, uma execução do programa é:

Digite um número de 4 dígitos: 5382
Seu número invertido é 2835

```
#include<stdio.h>
#include<cc50.h>

int
main(void)
{
        int num;
        int numinv;
        int digito;

        do
        {
        printf("Digite um número de 4 dígitos: ");
        num = GetInt();
        }
        while (num < 1000 || num > 9999);

        while (num > 0)
        {
        digito = num % 10;
        num /= 10;
        numinv = numinv * 10 + digito;
        }

        printf("Seu número invertido é: %d\n", numinv);
}
```

2. Desenvolva um programa que leia do terminal três valores inteiros positivos. Eles corresponderão a comprimentos de segmentos de retas. Verifique se eles formam um triângulo e em caso afirmativo classifique o triângulo. A saída do programa deve imprimir em letra minúscula:

'n' se não for possível formar um triângulo;
'a' se o triângulo formado for acutângulo;
'r' se o triângulo formando for retângulo;
'o' se o triângulo formado for obtusângulo.

```
#include<stdio.h>

int
main(void)
{
	int a, b, c;

	printf("Digite três valores inteiros positivos: \n");
	scanf("%d %d %d", &a, &b, &c);

	//Verifica se os valores formam um triângulo
	if (a + b > c && a + c > b && b + c > a)
	{
		//Cálculo dos quadrados dos lados
		int a_quad = a * a;
		int b_quad = b * b;
		int c_quad = c * c;

		//Verifica a classificação dos triângulos
		if (a + b > c && a + c > b && b + c > a)
		{
			if (a_quad + b_quad == c_quad || a_quad + c_quad == 
b_quad || b_quad + c_quad == a_quad)
				printf("\nr\n");		//Triang retângulo

			else if (a_quad + b_quad < c_quad || a_quad + c_quad < 
b_quad || b_quad + c_quad < a_quad)
				printf("\no\n");		//Triang obtusângulo

			else
				printf("\na\n");		//Triang acutângulo
		}
		else
			printf("\nn\n");			// Não forma triang
	}
}
```

3. Atualmente os carros são bicombustíveis, ou seja, é possível abastecer com gasolina ou etanol. No entanto, o desempenho dos carros é diferente em termos de distância por litro que se consegue percorrer utilizando um ou outro combustível. Construa um programa que, dados o preço do litro de etanol, o preço do litro de gasolina e os quilômetros por litro que um carro bicombustível realiza com cada um desses combustíveis, determine se é mais vantajoso abastecer com etano ou gasolina. No caso de empate a preferência é por gasolina, pois é mais fácil para ligar o carro de manhã.

Entrada: uma única linha contendo quatro números reais com precisão de duas casas decimais, P_E, P_G, R_E, R_G, representado respectivamente, o preço do etanol, o preço da gasolina, o rendimento (Km/l) do carro utilizando etanol e o rendimento (Km/l) do carro utilizando gasolina.

Saída: deve ser composta por uma única linha contendo o caractere “E” se é mais econômico abastecer com etanol ou o caractere ‘G’ se é mais econômico com gasolina (ou o caso de empate como colocado anteriormente).

```
#include<stdio.h>
int
main(void)
{
        float P_E, P_G, R_E, R_G;

        //Valores de entrada
        printf(Digite o preço do etanol, o preço da gasolina, o rendimento Km/L de etanol e o rendimento Km/L de gasolina, respectivamente: “);
        scanf(“%f %f %f %f”, &P_E, &P_G, &R_E, &R_G);

        //Cálculo do custo por km para cada combustível
        float custokmE = P_E / R_E;
        float custokmG = P_G / R_G;

        //Verificação do combustível mais vantajoso
        char vant;
        if (custokmE < custokmG || (custokmE == custokmG && R_G == R_E))
                vant = ‘E’
        else
                vant = ‘G’
```

printf(“\n%c\n, vant);

4. Faça um programa para calcular o Imposto de Renda de um usuário a partir do salário base que ele informará. A tabela da Receita Federal para o ano de 2021 está reproduzida a seguir:

Salário base (R$)	Alíquota
Até 1.903,98	Isento
1903,99 a 2.826,65	7,5%
2.826,66 a 3.751,05	15%
3.751,06 a 4.664,68	22,5%
Acima de 4.664,68	27,5%

Os percentuais de alíquota são aplicados a cada faixa salarial. O valor total pago é o acumulado a ser pago nas diferentes faixas.

O programa deve imprimir na tela uma única linha, contendo o valor do Imposto de Renda a ser recolhido.

```
#include<stdio.h>
#include<cc50.h>

float imposto;

printf(“Digite o seu salário: R$”);
float salario = GetFloat();

//Isento de imposto de renda
if (salario <= 1903.98)
printf(“Você está isento”);

else
{
        //7.5% de imposto
        if (salario <= 2826.65)
        imposto = (salario – 1903.98) * 0.075;

        //15% de imposto
        else if (salario <= 3751.05)
        imposto = (salario – 2826.65) * 0.15 + 142.80;

        //22.5% de imposto
        else if ()
        imposto = (salario – 3751.05) * 0.225 + 354.80;
```

```
        //27.5% de imposto
        else
        imposto = (salario – 4664.68) * 0.275 + 636.13;

        printf(“O Imposto de Renda a ser pago é: R$%.2f\n”, imposto);
        }
}
```

5. Escreva um programa que calcule o ângulo entre dois vetores no espaço bidimensional. O usuário lhe informará as coordenadas de cada um dos vetores.

```
#include<stdio.h>
#include<math.h>

#define PI 3.14159265

int
main(void)
{
        float x1, x2, y1, y2
        float comp1, comp2, prodesc, ang;

        printf(“Digite as coodernadas do vetor 1, sendo o primeiro valor X e o segundo Y: “);
        scanf(“%f %f”, &x1, &y1);

        printf(“Digite as coodenadas do vetor 2, sendo o primeiro valor X e o segundo Y: “);
        scanf(“%f %f”, &x2, &y2);

        //Cálculo do produto escalar
        prodesc = x1 * x2 + y1 * y2;

        //Cálculo dos comprimentos dos vetores
        comp1 = sqrt (x1 * x1 + y1 * y1);
        comp2 = sqrt (x2 * x2 + y2 * y2);

        //Cálculo do ângulo
        ang = acos (prodesc / (comp1 * comp2));

        //Transformação de rad para graus
        ang = ang * 180 / PI;

        printf(“\nO ângulo entre os vetores 1 e 2 é de %.2f\n”, ang);
```

```
}
```