

Lista de exercícios (GDB)

Instalação e Configuração:

- Instale o GCC em seu sistema operacional;
- Instale o GDB em seu sistema operacional;
- Compile um código simples em linguagem C com depuração (usando a flag -g);
- Coloque um breakpoint em uma determinada linha de código e execute o programa até esse ponto.
- Execute o programa no GDB usando o comando *run*;

Código 1 - Fibonacci

```
1 #include<stdio.h>
2 int fibonacci(int n) {
3     if (n <= 1) {
4         return n;
5     }
6     else {
7         return fibonacci(n - 1) + fibonacci(n - 2);
8     }
9 }
10 int main(){
11     fibonacci(15);
12 }
```

- Depure o código para descobrir quantas vezes o **fibonacci(5)** é calculado (requisitado).
- Descubra através da depuração, qual a maior quantidade de níveis (*frames*) esse código alcança.
- Depure o código até o momento que o primeiro fibonacci(14) é calculado e o executável irá começar a calcular o fibonacci(13).

Código 2 - Operação Matemática

```
1  #include <stdio.h>
2
3  // Funcao para calcular algo
4  int operacaoMatematica(int a, int b) {
5      int r;
6
7      while (b != 0) {
8          r = a % b;
9          a = b;
10         b = r;
11     }
12
13     return a;
14 }
15
16 int main() {
17     int num1, num2;
18
19     printf("Digite o primeiro numero: ");
20     scanf("%d", &num1);
21
22     printf("Digite o segundo numero: ");
23     scanf("%d", &num2);
24
25     int res = operacaoMatematica(num1, num2);
26
27     printf("A operacao de %d e %d e: %d\n", num1, num2, res);
28
29     return 0;
30 }
```

- Execute o algoritmo em modo de depuração.
- Descubra através da depuração, qual a maior profundidade de (*frames*) esse código alcança.
- Descubra o que ocorre quando um parâmetro é 0.
- Descubra o que ocorre quando os dois parâmetros apresentados são iguais e maiores que 1.

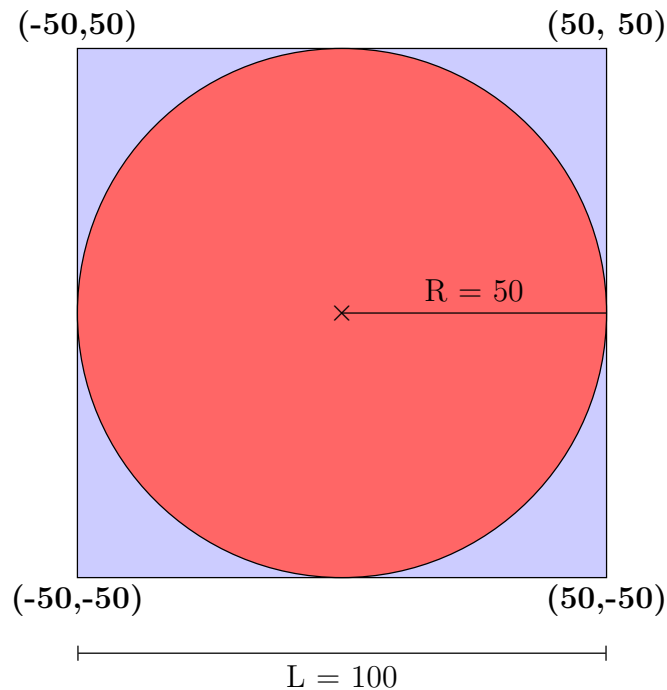
Código 3 - Números primos

```
1  #include <stdio.h>
2
3  // Funcao recursiva para verificar se um numero e primo
4  int ehPrimoRecursivo(int n, int i) {
5      if (n <= 2) {
6          return (n == 2);
7      }
8
9      if (n % i == 0) {
10         return 0;
11     }
12
13     if (i * i > n) {
14         return 1;
15     }
16
17     return ehPrimoRecursivo(n, i + 1);
18 }
19
20 // Funcao para imprimir numeros primos em um intervalo
21 void imprimirPrimosIntervalo(int inicio, int fim) {
22     if (inicio > fim) {
23         return;
24     }
25
26     if (ehPrimoRecursivo(inicio, 2)) {
27         printf("%d ", inicio);
28     }
29
30     imprimirPrimosIntervalo(inicio + 1, fim);
31 }
32
33 int main() {
34     int inicio, fim;
35
36     printf("Digite o inicio do intervalo: ");
37     scanf("%d", &inicio);
38
39     printf("Digite o fim do intervalo: ");
40     scanf("%d", &fim);
41
42     printf("Numeros primos entre %d e %d: ", inicio, fim);
43     imprimirPrimosIntervalo(inicio, fim);
44     printf("\n");
45
46     return 0;
47 }
```

- a) Através da depuração, compreenda como o código funciona.
- b) Desenhe em formato de uma árvore de recursão os passos para o cálculo da procura dos números primos de 10 a 20 (usando a depuração).
- c) Conte a quantidade de chamadas recursivas para realizar o resultado de b).
- d) Depure o algoritmo com os valores de entrada de 1000 e 10000. Utilize o depurador para interromper a depuração quando o 240º número primo aparecer.

Código 4 - Função aleatória

A figura a seguir representa um círculo inscrito em uma circunferência. A origem (0,0) do centro de coordenadas coincide com a origem do círculo e o centro do quadrado. Foi desenvolvido um algoritmo para gerar pontos aleatórios que aparecerão nos limites deste quadrado. O objetivo é selecionar apenas os pontos que estão dentro do círculo. A seguir, apresentamos o código e algumas questões relacionadas ao mesmo.



```
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int randomInRange(int min, int max) {
5      return min + rand() % (max - min + 1);
6  }
7
8  void gerarNumerosAleatorios() {
9      int i;
10     for (i = 0; i < 100; i++) {
11         int px = randomInRange(-50, 50);
12         int py = randomInRange(-50, 50);
13         printf("P = (%d, %d)\n ", px, py);
14     }
15     printf("\n");
16 }
17
18 int main() {
19     srand(42);
20     gerarNumerosAleatorios();
21 }
```

- a) Através da depuração, realize a interrupção do algoritmo toda vez que as duas coordenadas são maiores que 0.
- b) Utilize a depuração para realizar interrupções no algoritmo somente quando os pontos estiver na região externa ao círculo.
- c) Utilize a depuração para realizar interrupções no algoritmo somente quando os pontos estiver na região interna ao círculo.