

## Uso de algumas funções das bibliotecas stdlib.h e time.h

A função `rand()` está em `stdlib.h` e devolve um número pertencente à faixa `0..RAND_MAX`.

Fazendo a divisão desse número por `RAND_MAX+1` obtemos um número `d` tal que  $0 \leq d < 1$ .

Para mapear o número `d` no intervalo `[a,b]` em que `a, b` são inteiros, podemos fazer:

- 1) A multiplicação de  $(b-a+1)$  por `d` (`d` é número real maior ou igual a 0 e menor do que 1) produz como resultado um número cuja parte inteira pertence à faixa de números inteiros `0,1,2,3...b-a`.
- 2) A soma desse número inteiro (parte inteira do resultado anterior) com o número `a` resulta um número pertencente à faixa de inteiros de `a` até `b`.

$$k = d \times (b - a + 1) \text{ e } 0 \leq d < 1 \Rightarrow k \in \{0,1,2,\dots,b-a\}$$

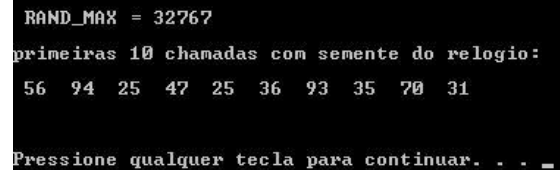
$$g = a + k \Rightarrow g \in [a,b]$$

A função `rand` faz cálculos com um valor inicial para produzir o primeiro número. Em seguida, utiliza o resultado anterior para gerar o próximo número e assim por diante.

O valor inicial utilizado é denominado semente e podemos escolher a semente por meio da função `srand(int)`.

Por exemplo, fazendo `srand((int)time(NULL));` utilizamos o relógio do sistema para definir a semente.

O código<sup>1</sup> a seguir exemplifica a utilização das funções usadas para gerar números pseudoaleatórios:



```
RAND_MAX = 32767
primeiras 10 chamadas com semente do relógio:
56 94 25 47 25 36 93 35 70 31
Pressione qualquer tecla para continuar. . . _
```

```
#include <stdio.h>
#include <stdlib.h>           // aqui estão as funções rand, srand, time
#define N 10                 // a tabela poderá armazenar até 10 números

main() {
    printf("\n\n RAND_MAX = %d \n\n", RAND_MAX);
    int tab[N]; //o array tab armazena os números gerados aleatoriamente
    int i,k,g,a,b;
    double d;
    a = 1; b = 100; // os números serão mapeados em números de 1 a 100
    srand((int) time(NULL)); //semente obtida pelo relógio do sistema

    printf("primeiras 10 chamadas com semente do relógio: \n\n");

    // o resultado em double aumenta a quantidade de dígitos
    // k "pega" os dois primeiros algarismos na parte decimal
    // armazenamos cada número no array tab

    for(i=0; i < N; i++){
        d = (double)rand()/((double)RAND_MAX + 1);
        k = (int)(d*(b-a+1));
        g = a+k;
        tab[i] = g;
    };
    for (i=1;i<N;i++) printf(" %d ", tab[i]);
    printf("\n\n"); system("PAUSE");
}
```

Na biblioteca `time.h` estão as declarações de tipos e as funções para acessar o relógio do sistema. São duas funções que podemos utilizar:

- 1) A função<sup>2</sup> `time()` devolve o tempo decorrido desde uma data inicial (janeiro de 1970) até a data corrente.
- 2) A função<sup>3</sup> `clock()` devolve o número de tics do relógio do computador desde o início da execução do trecho do programa.

<sup>1</sup> `gerarAleatorioInteiro.c`

<sup>2</sup> `testeTIME.c`

<sup>3</sup> `testeCLOCK.c`

### Veja o código (com o uso de CLOCK):

```
#include <time.h> // aqui estão as declarações de tipos e as funções de
                  // manipulação de data e tempo

main(){
    int f;
    double pf;
    clock_t inicio, fim; // variáveis do tipo clock_t para os dados
    printf(" \n\n");      printf("\n aguarde o tempo passar... \n");

    //PONTO A
    inicio = clock(); // inicio = número de tics (tempo de CPU) desde o início da execução
    for(f=1;f<=1000000000;f++);
    printf("\n quase 4 segundos...\n\n");
    //PONTO B
    fim = clock(); // fim = número de tics desde o início da execução

    pf = (double)(fim - inicio)/CLOCKS_PER_SEC; // pf = número de tics no intervalo de A até B.

    printf("Tempo medido em quantidade de tics = %.8f \n",pf);
    printf(" \n\n "); system("PAUSE");
}
```

### Veja o código (com o uso de TIME):

```
#include <time.h> // aqui estão as declarações de tipos e as funções de
                  // manipulação de data e tempo

main(){
    int f;
    double d;
    struct tm *ptr;          // struct tm guarda os dados do calendário ptr é um pointer para tm
    time_t inicio, fim;

    printf(" Data e hora corrente: ");
    inicio = time(NULL);
    ptr = localtime(&inicio); // para pegar o calendário do momento
    printf(asctime(ptr));      // e exibir a data e hora na tela
    printf("\n Quantidade de segundos desde janeiro de 1970 = %d \n",inicio);

    printf(" \n\n"); printf("\n  Aguarde o tempo passar... ");

    inicio = time(NULL); //tempo corrente em uma variável do tipo time_t
    //PONTO A
    for(f=1;f<=1000000000;f++);
    printf(" aproximadamente 3 segundos...\n\n");
    //PONTO B
    fim = time(NULL);

    d = difftime(fim,inicio);
    printf("    Tempo de execucao = %f segundos \n\n",d);

    inicio = time(NULL);

    printf("\n\n Data e hora corrente: ");
    ptr = localtime(&inicio); // para pegar o calendário do momento
    printf(asctime(ptr));      // e exibir a data e hora na tela
    printf("\n Quantidade de segundos desde janeiro de 1970 = %d \n\n",inicio);

    printf(" \n\n ");  system("PAUSE");
}
```