

TAD Timer

Vamos construir um TAD para nos auxiliar a medir o tempo de execução de trechos de código. Existem diferentes formas para se fazer isso. Veja nesse [LINK](#) o artigo "*8 Ways to Measure Execution Time in C/C++*", algumas alternativas.

A ideia de criar esse TAD é abstrair a complexidade de como fazer e facilitar a vida do programador que precisa dessa medição frequentemente.

Dados

Quais dados precisam ser representados e manipulados?

- Tempo gasto em segundos ou milissegundos
 - O que nos interessa é o tempo final, mas para isso provavelmente precisaremos anotar o tempo de início e o tempo de término para depois computarmos o tempo gasto.

Operações

Quais funcionalidades serão disponibilizadas?

1. Criar o timer
2. Desalocar o timer
3. Iniciar o timer
4. Parar o timer
5. Limpar o timer (reset)
6. Obter o tempo gasto

Especificação

```
tad_timer.h
```

```

/*****
* DADOS
*****/

typedef struct timer Timer;
/*****
* OPERACOES
*****/

/** * Cria um timer */ Timer* timer_criar(); /** * Destroi um ti
*/ void timer_desalocar(Timer** t); /** * Inicia o timer */ void
timer_start(Timer* t); /** * Para o timer */ void timer_stop(Tim
t); /** * Reinicia o timer */ void timer_reset(Timer* t); /** *
Devolve o resultado. * Caso o timer ainda não tenha finalizado
devolve -1 */ float timer_resultado(Timer* t);

```

Utilização

Medir o tempo de execução de um trecho de código que realiza
1.000.000.000 de somas e divisões.

main.c

```
#include <stdio.h> #include "tad_timer.h" int main(){ Timer*
t1 = timer_criar(); /** * Medir o tempo de execução de um
trecho de código que realiza * 1.000.000.000 de somas e
divisões */ long int soma = 0; double divisao = 1; int
interacoes = 1000*1000*1000; printf("Executando %d de
operacoes\n", interacoes); timer_start(t1);
//----- for (int
i=0; i<interacoes; i++) { soma += i; divisao /= 2; }
//-----
timer_stop(t1); printf("Tempo decorrido: %.2f\n\n",
timer_resultado(t1)); /** * Medir o tempo de percorrer uma
matriz */ long int linhas = 100000; long int colunas =
100000; printf("Executando %ld de operacoes\n",
linhas*colunas); timer_reset(t1); timer_start(t1);
//----- long int
i,j; for(i=0; i < linhas; i++){ for(j=0; j < colunas; j++){
// leitura da matriz } }
//-----
timer_stop(t1); printf("Tempo decorrido: %.2f\n",
timer_resultado(t1)); timer_desalocar(&t1); }
```

Implementação

a) Desenvolva 2 implementações nos arquivos `tad_timer1.c` e `tad_timer2.c` utilizando os métodos 4 e 5 do artigo [*8 Ways to Measure Execution Time in C/C++*](#).

`tad_timer1.c`

```
#include "tad_timer.h"
```

`tad_timer2.c`

```
#include "tad_timer.h"
```

b) Crie o arquivo `makefile` para automatizar a compilação e execução. Como teremos 2 implementações, utilize o seguinte padrão.

Para compilação e execução da implementação `tad_timer1.c`

```
make timer1
```

Para compilação e execução da implementação `tad_timer2.c`