

1 Sobre a entrega do trabalho

São requisitos para atribuição de notas a este trabalho:

- Uso de um arquivo **makefile** para facilitar a compilação. Os professores rodarão “make” e deverão obter o arquivo executável funcional com a sua solução. Este executável deverá estar no subdiretório **tp3e** deve ter o nome (**tp3**);
- Opções de compilação: deve incluir **-Wall** e **-std=c90**. Haverá desconto na nota se compilador mostrar algum “warning”;
- Seu programa deve desalocar (**free**) toda a memória dinâmica alocada, isto será conferido pelo uso do programa **valgrind**, cada erro também vai gerar desconto na nota;
- Arquivo de entrega:
 - deve estar no formato tar comprimido e deverá obrigatoriamente ter o nome (**tp3.tar.gz**);
 - O **tar** deve ser criado considerando-se que existe um diretório com o nome do trabalho. Por exemplo, este trabalho é o **tp3**;
 - Então seu **tar** deve ser criado no diretório pai do subdiretório **tp3**, o qual deve conter todos os arquivos que serão entregues (**tar zcvf tp3.tar.gz tp3**), de maneira que os professores, ao abrirem o **tar** com o comando “**tar zxvf tp3.tar.gz**” obterão um diretório **tp3** com as suas respostas;
 - Os professores testarão seus programas em uma máquina do departamento de informática (por exemplo, **cpu1**), por isso, antes de entregar seu trabalho faça um teste em máquinas do **dinf** para garantir que tudo funcione bem.

2 Objetivos

São objetivos deste trabalho a prática dos seguintes conceitos:

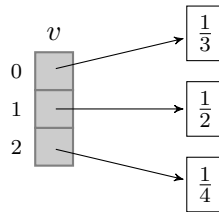
- Alocação dinâmica de structs e de vetores;
- Manipulação de ponteiros;
- Uso de números pseudo-aleatórios;
- Uso da ferramenta **valgrind**.

3 O trabalho

Você deve implementar um programa que manipule ponteiros para números racionais, que são números da forma $\frac{num}{den}$, onde num e den são números inteiros.

Inicialmente, você vai alocar dinamicamente um vetor de ponteiros para números racionais. Em seguida, você vai gerar vários ponteiros para números racionais gerados aleatoriamente e vai inserir estes ponteiros em ordem no vetor.

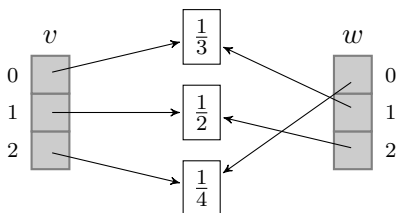
A título de exemplo, considere a figura abaixo. Pode-se ver um vetor v contendo três elementos (índices de 0 a 2). O exemplo mostra que foram lidos, nesta ordem, os números racionais $\frac{1}{3}$, $\frac{1}{2}$ e $\frac{1}{4}$, os quais foram inseridos respectivamente nas posições 0, 1 e 2 do vetor.



Agora seu programa deve criar um segundo vetor, digamos w , que vai conter ponteiros para os racionais ordenados em ordem crescente.

A ideia é que a *struct* pode ser grande e não queremos ficar trocando estas de lugar, só queremos movimentar ponteiros, que custa bem menos.

Veja na figura abaixo que o vetor v não foi alterado, mas que agora o vetor w , quando percorrido do índice 0 até o índice 2, permite ver os racionais ordenados, isto é, as posições 0, 1 e 2 do vetor w apontam respectivamente para os racionais $\frac{1}{4}$, $\frac{1}{3}$ e $\frac{1}{2}$, isto é, estão ordenados.



4 Seu programa

Você vai receber junto com este enunciado um arquivo `tp3.tar.gz` o qual depois de aberto conterá três arquivos, sendo um deles o arquivo `lib_racionais.h` que deve ser utilizado como base para a construção de dois outros arquivos:

- `lib_racionais.c`;
- `tp3.c`.

O arquivo `lib_racionais.h` não poderá ser modificado por você, na correção os professores usarão o mesmo arquivo que foi disponibilizado. Seu arquivo `lib_racionais.c` deve ser implementado considerando os protótipos na construção dos outros dois arquivos. Note que as funções devem funcionar exatamente como está escrito nos comentários delas.

Caso você precise, ou queira, usar outras funções, estas devem estar localizados em um dos outros dois arquivos, a escolha do local correto faz parte da avaliação.

Você também vai receber uma sugestão de arquivo `makefile` e uma outra sugestão de implementação do arquivo `tp3.c`, estes podem ser modificados a vontade.

Seu arquivo `tp3.c` deve usar a sua `lib_racionais.h` para resolver o problema descrito acima, que é melhor detalhado abaixo.

- Defina um nome `MAX`, que pode ter um valor 100, por exemplo;
- Crie uma função que gere aleatoriamente um número racional e retorna um ponteiro para ele. use a função `rand()` para isso. Note que uma única vez no seu programa você deve inicializar a semente randômica com o comando `srand(0)` (para facilitar a depuração).
- Crie uma função que gere um vetor de ponteiros para os números gerados, na ordem em que eles forem gerados. Sua função deve retornar este vetor. Este vetor deve ter o tamanho `MAX`;

- Crie uma função que rearrange os ponteiros deste segundo vetor de maneira que ao percorrê-lo do início ao fim, tenhamos os racionais ordenados;
- Crie uma função que receba um vetor de ponteiros para racionais e imprima os racionais na ordem em que eles são apontados pelo vetor do parâmetro. Assim, ao percorrer o primeiro vetor no exemplo acima a saída deve ser:

1/3 1/2 1/4

Mas ao se percorrer o segundo vetor a saída deve ser:

1/4 1/3 1/2

- Sua função de impressão deve imprimir os racionais separados por espaço em branco e após o último número não pode haver nenhum espaçamento antes do `\n`;
- Os números resultantes devem ser na forma simplificada, assim, por exemplo, se um cálculo resultar em $\frac{12}{9}$ então o retorno deve ser $\frac{4}{3}$. Você pode implementar uma função que simplifica um número racional para facilitar;
- Assim como no `tp1`, se um racional tiver denominador 1 deve ser impresso somente o numerador, bem como se o racional for zero, deve ser impresso somente 0.

Exemplo de entrada para `n=15` e a saída correspondente (lembre que os números são aleatórios e portanto sua saída pode ter números diferentes)

15

1/15 2/11 1/5 1/2 1/2 7/13 3/5 9/13 5/6 10/11 7/6 11/8 3 14/3 6

Bom trabalho!