Claudio Rodrigo Gish – 228366
Marcos Henrique Backes – 228483

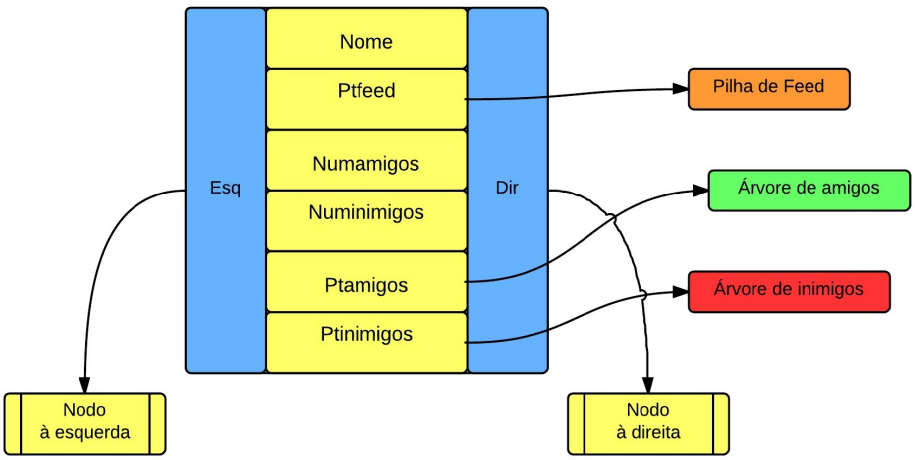
Trabalho Final de Estruturas de Dados

Like✓Unlike*x*.NET – Estruturas de dados utilizadas

Tipo usuário

Tipo usuário é a estrutura de dados utilizada para armazenar todas as informações de um usuário cadastrado na rede:

Tipo Usuário



Nome: string de até 100 caracteres que armazena o nome do usuário.

Ptfeed: ponteiro do tipo feed que aponta para a pilha de posts visíveis ao usuário.

Numamigos: tipo int, representa o número de vezes que o usuário foi adicionado como amigo por um outro usuário da rede.

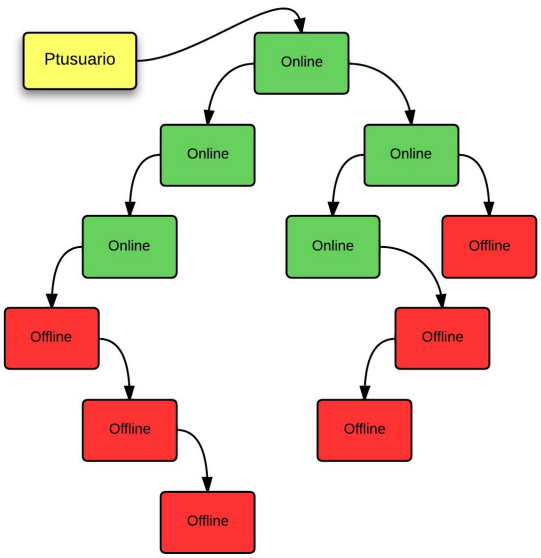
Ptamigos: ponteiro do tipo amigo/inimigo, aponta para a árvore de amigos do usuário.

Ptinimigos: ponteiro do tipo amigo/inimigo, aponta para a árvore de inimigos do usuário.

Esq: ponteiro do tipo usuário que aponta para o nodo à esquerda.

Dir: ponteiro do tipo usuário que aponta para o nodo à direita.

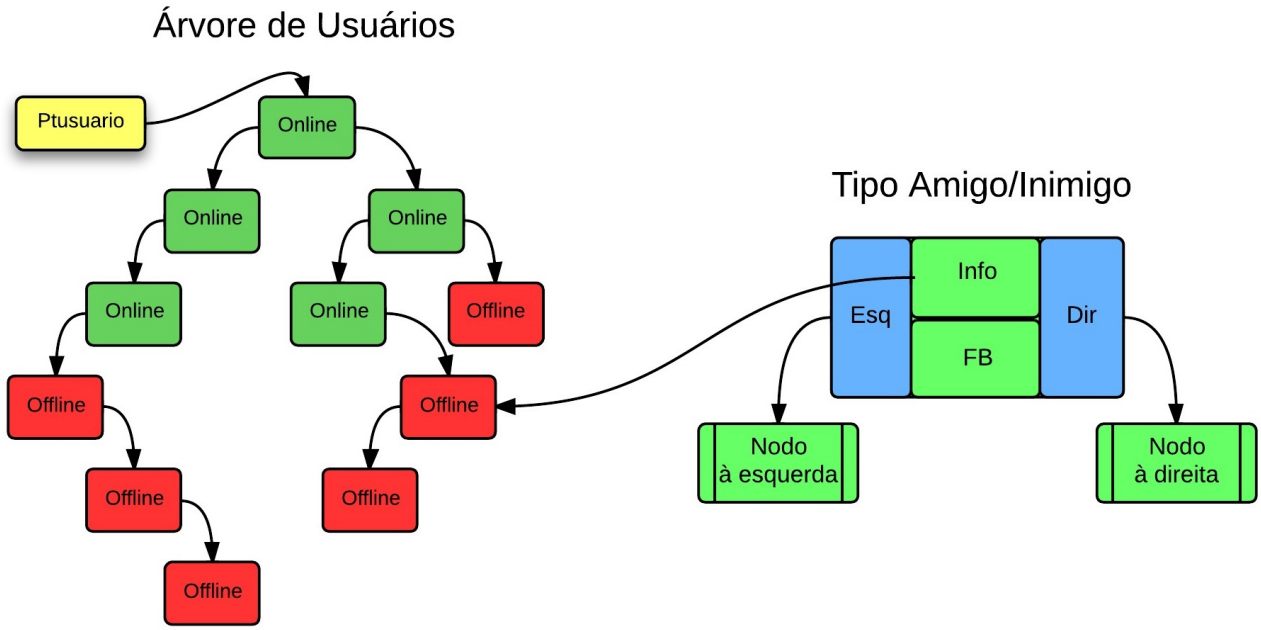
Árvore de Usuários (splay)



Os usuários são organizados em uma árvore splay, em ordem alfabética de nome. O principal motivo para a escolha dessa árvore é o fato de que todas as operações que um usuário pode fazer na rede (Inserir amigo, Escrever post, etc) precisam fazer uma consulta ao usuário na rede. Dessa forma, sempre que um usuário realiza uma operação, seu nodo é movido para o topo da árvore. Ao parar de utilizar a rede, seu nodo vai para a parte de baixo da árvore. Logo, a parte de cima da árvore estarão os usuários online e na parte de baixo os usuários offline. A desvantagem é que o primeiro acesso do usuário na sessão será mais lento, enquanto que a vantagem é que os acessos seguintes serão mais rápidos.

Tipo amigo/inimigo

Tipo amigo/inimigo é a estrutura utilizada para armazenar os amigos de um usuário da rede:

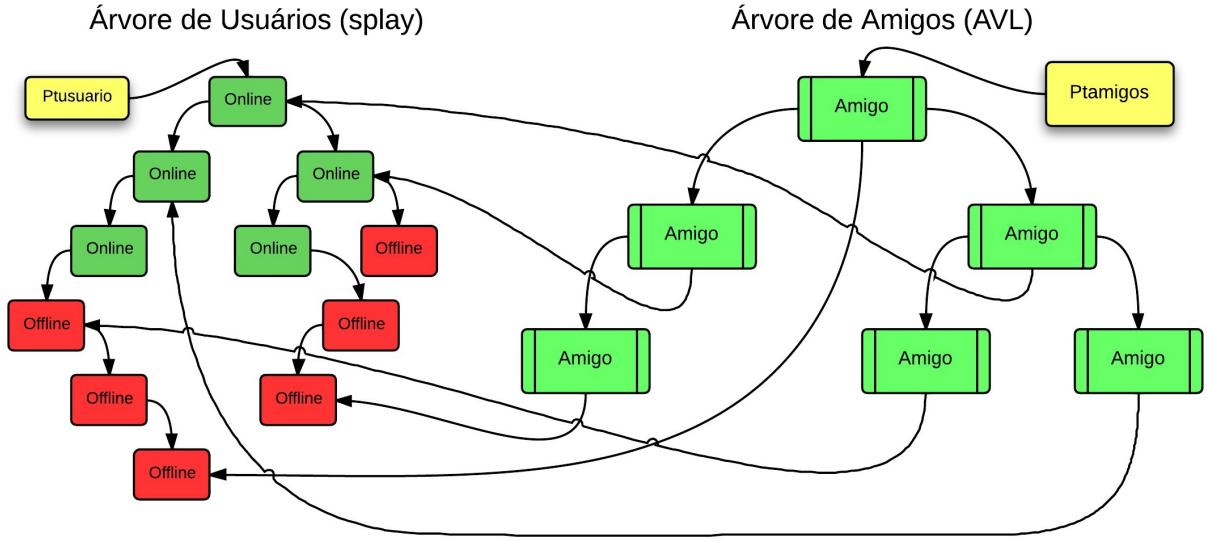


Info: ponteiro do tipo usuário que aponta para um usuário da rede. Esse usuário é o amigo/inimigo.

FB: tipo int, representa o fator de balanceamento (AVL).

Esq: ponteiro do tipo amigo/inimigo que aponta para o nodo à esquerda.

Dir: ponteiro do tipo amigo/inimigo que aponta para o nodo à direita.

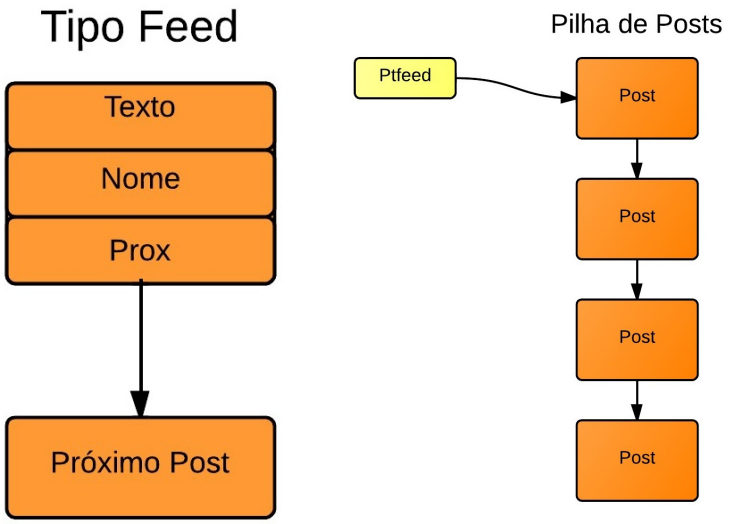


Os amigos de um usuário são organizados em uma árvore AVL, por ordem alfábetica do nome do usuário apontado por info. A desvantagem dessa árvore é que as inserções de novos amigos são muito demoradas. No entanto, como são necessárias muitas consultas na árvore de amigos para as operações de post e como a operação de consulta é muito rápida na árvore AVL, essa estrutura de dados foi a escolhida.

Os inimigos são organizados da mesma forma que os amigos.

Tipo feed

Tipo feed é a estrutura utilizada para armazenar os posts que são visíveis a um usuário:



Texto: string de até 80 caracteres que armazena o texto postado.

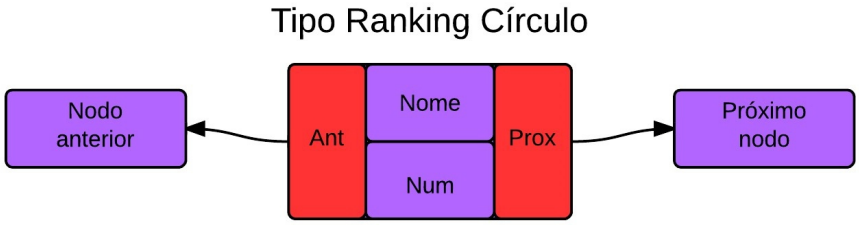
Nome: string de até 100 caracteres que armazena o autor do texto.

Prox: ponteiro do tipo feed que aponta para o próximo post.

A estrutura escolhida para armazenar os posts foi uma pilha pelo fato de que só são feitas insersões no topo. Assim, os posts novos ficam no topo e os antigos no fundo. Quando um usuário escreve um post, ele insere o post na pilha de seus amigos, inimigos ou todos os usuários. Para exibir os posts dos amigos ou inimigos de um usuário, a pilha de posts é percorrida. Para cada post, é verificado se o autor está na lista de amigos ou inimigos do usuário (depende do tipo de operação). Se estiver, o post é impresso, senão, não é impresso. A vantagem é que são necessárias menos consultas para inserir e exibir os posts, mas a desvantagem é que esse armazenamento ocupa mais espaço na memória.

Tipo ranking círculo

Essa estrutura é usada para armazenar temporariamente o ranking popular da “Operação Ranking Amigo”.

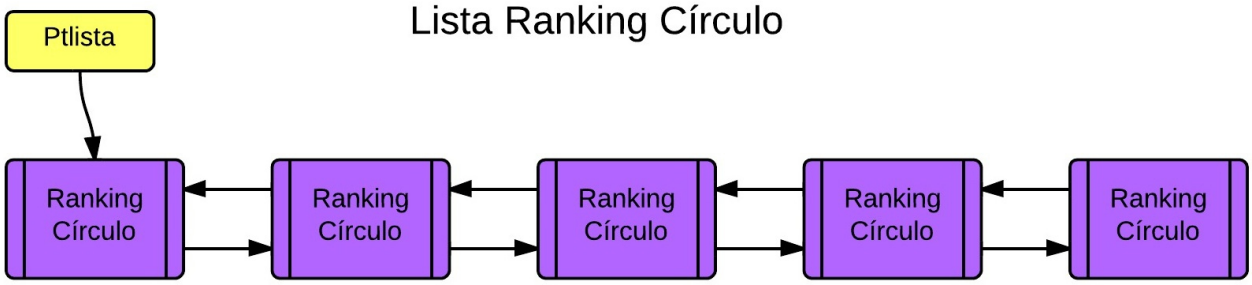


Nome: string de até 100 caracteres que armazena o nome do usuário.

Num: tipo int, representa o número de vezes que o usuário aparece na lista de amigos, inimigos ou ambos dos amigos de um usuário.

Ant: ponteiro do tipo ranking círculo que aponta para o nodo anterior.

Prox: ponteiro do tipo ranking círculo que aponta para o próximo nodo.



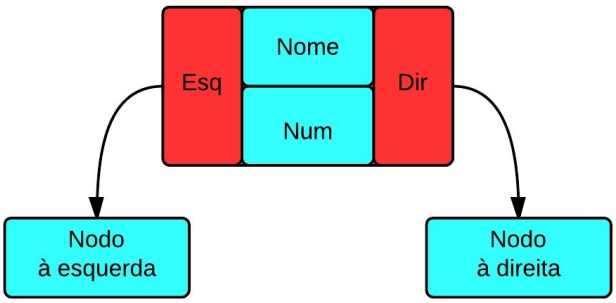
A estrutura escolhida para armazenar esses dados foi uma lista duplamente encadeada em ordem decrescente do valor do campo num e em segundo lugar, em ordem alfabética da string nome. Para criar essa lista, o programa percorre as listas de amigos dos amigos de um usuário. Se o nome do usuário não estiver na lista, esse nome é inserido no final com o campo num valendo 1, depois reordena, se necessário, os nodos cujo campo num vale 1 em ordem alfabética. Se o nome do usuário estiver na lista, o campo num é incrementado, se necessário, reordena os nodos pela ordem numérica, passando esse nodo à frente dos nodos com campo num menor, e, depois, pela ordem alfabética, ordena os nodos com o mesmo valor de num em ordem alfabética. Para imprimir o ranking basta percorrer a lista.

A vantagem de escolher essa estrutura é que são necessárias muitas trocas para criar essa lista e é muito fácil trocar dois nodos de lugar em uma lista duplamente encadeada. A desvantagem é que para inserir um nodo novo, é necessário percorrer toda a lista.

Tipo ranking geral

Tipo ranking geral é a estrutura usada para armazenar o ranking gerado pela “Operação Ranking Geral”:

Tipo Ranking Popular



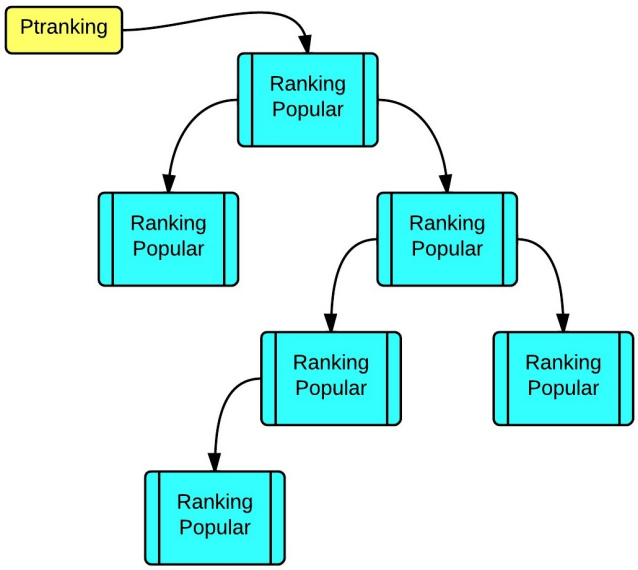
Nome: string de até 100 caracteres que armazena o nome do usuário.

Num: tipo int, representa o número de vezes que o usuário aparece na lista de amigos, inimigos ou ambos de todos os usuários.

Esq: ponteiro do tipo ranking geral que aponta para o nodo à esquerda.

Dir: ponteiro do tipo ranking geral que aponta para o nodo à direita.

Árvore Ranking Popular (ABP)



A estrutura escolhida foi uma ABP em ordem crescente do campo num. Como o número de vezes que um usuário aparece nas listas de amigos ou inimigos de todos os usuários, não é necessário percorrer todas as listas de amigos para criar essa lista, por isso não são necessárias trocas de lugar. Nesse caso é vantagem a escolha da ABP porque não são necessárias consultas, o principal é inserir os nodos de forma rápida. Para imprimir o ranking, é só percorrer a árvore com o caminhamento central à direita.