INSTITUTO DE ENGENHARIA E TECNOLOGIA - IET *Curso: Ciência da Computação*	
Disciplina: Programação e Estruturas de Dados	
Professora: Jaqueline Faria de Oliveira	**Valor: 10 pontos**

Instruções:

- **Trabalho pode ser feito no máximo em dupla;**
- **Data de entrega: 30/06/2017 até 23h59;**
- **Deve ser enviado somente o arquivo .cpp contendo o código do programa;**
- **No cabeçalho do arquivo deve conter um comentário com o nome e matrícula dos integrantes da dupla;**
- **O arquivo deve ser enviado via SOL > Sala de Aula Virtual;**
- **Serão avaliados:**
  - **Identação ;**
  - **Clareza do código;**
  - **Utilização das estruturas solicitadas;**
  - **Resolução correta do problema proposto;**
  - **Documentação do algoritmo através de comentários.**
- **Atenção: Qualquer identificação de cópia e/ou plágio os trabalhos que estiverem nessa situação receberão nota 0 (zero)**

TRABALHO PRÁTICO 3

1. Cache web – Histórico de Acessos na Internet

Em redes de computadores, um proxy (em português procurador) é um servidor (um sistema de computador ou uma aplicação) que age como um intermediário para requisições de clientes solicitando recursos de outros servidores. Um cliente conecta-se ao servidor proxy, solicitando algum serviço, como um arquivo, conexão, página web ou outros recursos disponíveis de um servidor diferente e o proxy avalia a solicitação como um meio de simplificar e controlar sua complexidade. Os proxies foram inventados para adicionar estrutura e encapsulamento a sistemas distribuídos. Hoje, a maioria dos proxies são proxies web, facilitando o acesso ao conteúdo *na World Wide Web* e fornecendo anonimato.
Uma aplicação *proxy* popular é o *proxy* de armazenamento local (ou *cache*) web, em inglês *caching web proxy*, um *proxy web* usado para armazenar e atualizar (conforme pré-programado). Este provê um armazenamento local de páginas da Internet e arquivos disponíveis em servidores remotos da Internet assim como sua constante atualização, permitindo aos clientes de uma rede local (LAN) acessá-los mais rapidamente e de forma viável sem a necessidade de acesso externo.
Quando este recebe uma requisição para acesso a um recurso da Internet (a ser especificado por uma URL), um proxy que usa cache procura resultados da URL em primeira instância no armazenamento local. Se o recurso for encontrado, este é consentido imediatamente. Senão, carrega o recurso do servidor remoto, retornando-o ao solicitante que armazena uma cópia deste na sua unidade de armazenamento local. O cache usa normalmente um algoritmo de expiração para a remoção de documentos e arquivos de acordo com a sua idade, tamanho e histórico de acesso (previamente programado). Dois algoritmos simples são o *Least Recently Used* (LRU) e o *Least Frequently Used* (LFU). O LRU remove os documentos que passaram mais tempo sem serem usados, enquanto o LFU remove documentos menos frequentemente usados.

2. Descrição do trabalho prático

Deve ser criado um programa que simule o funcionando de um histórico de Páginas de um Cache. Considere que o Cache tem tamanho máximo de 10 elementos no seu Histórico de Acessos. Quando esse histórico está cheio uma das páginas anteriores deve ser excluída. Devem ser implementados dois Históricos, um deve obedecer às regras de exclusão LRU e outro com regras LFU.
O usuário deve inserir manualmente as páginas para essa simulação. Podem ser inseridas quantas páginas forem necessárias, podendo estas ser repetidas ou não. A cada vez que uma página for informada o programa deve enviá-las para os dois Históricos e depois exibi-los na tela em ordem decrescente de data de acesso.

INSTITUTO DE ENGENHARIA E TECNOLOGIA - IET *Curso: Ciência da Computação*	
Disciplina: Programação e Estruturas de Dados	
Professora: Jaqueline Faria de Oliveira	**Valor: 10 pontos**

Ao enviar para cada histórico deve-se verificar:

- Histórico com regras LRU:
  - Se página já está no histórico, atualizar data de acesso.
  - Ao atualizar a página esta deve se tornar a primeira do histórico.
  - Se página não está no histórico, inserir essa página.
  - Caso o histórico esteja cheio, deve-se excluir um item do histórico conforme regras do algoritmo LRU.
- Histórico com regras LFU:
  - Se página já está no histórico, atualizar a quantidade de acessos.
  - Ao atualizar a página esta deve se tornar a primeira do histórico.
  - Se página não está no histórico, inserir essa página.
  - Caso o histórico esteja cheio deve-se excluir um item do histórico conforme regras do algoritmo LFU.

Devem ser dadas ao usuário as seguintes opções:

1. Inserir página no histórico
2. Listar Histórico LRU
3. Listar Histórico LFU
4. Limpar históricos
5. Sair

Implementação:

Crie um algoritmo que implemente uma estrutura de Página:

Estrutura Pagina:

- ID - Inteiro
- Título da página – Caractere de 100 posições
- URL – Caractere de 300 posições
- Data Acesso – Double – Valor em milissegundos obtido conforme anexo 1.
- Quantidade de Acessos – Inteiro

Deve ser implementado um **Tipo Abstrato de Dados Pilha de Páginas.** Deve-se utilizar a estrutura de Pilha com ponteiros vista em sala de aula. Devem ser utilizadas duas pilhas para implementação, uma dos históricos LRU e para os históricos LFU.

Regras para implementação das pilhas:

- A pilha deve se manter sempre organizada por data de acesso. Desta forma, quando uma página já faz parte do Histórico e é acessada novamente deve-se colocar a página no topo da pilha e atualizar sua data de acesso. Isso deve funcionar para ambas as pilhas.
- Demais regras de implementação devem obedecer às regras do Tipo Abstrato de Dados Pilha passado em sala (LIFO).

Devem ser feitas as seguintes validações:

- Não deve ser possível inserir dois históricos com mesmo ID

Bom trabalho a todos! ☺

Professora Jaqueline

INSTITUTO DE ENGENHARIA E TECNOLOGIA - IET *Curso: Ciência da Computação*	
Disciplina: Programação e Estruturas de Dados	
Professora: Jaqueline Faria de Oliveira	**Valor: 10 pontos**

Anexo 1:
Função para obtenção de tempo atual em milissegundos:
Tipo **time_t :** Um inteiro com sinal (32 ou 64 bits) que armazena o número de segundos desde a criação do Unix: meia-noite de 1 de Janeiro de 1970 UTC (com exceção dos segundos bissextos).

```
#include <stdlib.h>
#include <stdio.h>
#include <time.h>

int main(void)
{
	//Variável do tipo tempo
	time_t segundos;
	//Retorna o tempo em segundos
	segundos= time(NULL);
	printf("%d", segundos);
}
```