UFFS – Universidade Federal da Fronteira Sul
Curso de Ciencia da Computação
Organização de Computadores

Instruções:

Trabalho individual ou em dupla
Cronograma:

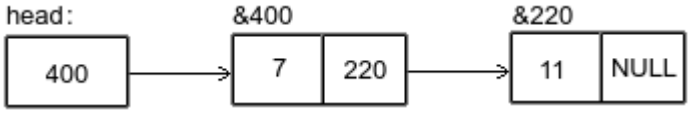
	em dia	atrasada
Entrega inicial:	02/06/2024	06/06/2024
Entrega intermediária:	12/06/2024	16/06/2024
Entrega final:	**23/06/2024**	30/06/2024
Apresentação:	**05/07/2024**	-

O que entregar:

Um arquivo com extensão .s ou .asm
O arquivo deve conter obrigatoriamente um cabeçalho com a matricula e o nome dos integrantes do trabalho

Descrição:

Faça um programa que implementa a gerencia de uma lista ordenada com capacidade de armazenar números inteiros.
O inicio da lista deve estar armazenado em uma posição de memória de nome *head* com valor inicial 0 (NULL).
Cada posição da lista ocupa 8 bytes, onde 4 bytes são usados para armazenar o valor inteiro e 4 bytes para armazenar o ponteiro para o próximo elemento. Por exemplo:



- O elemento encontra-se na posição de memória 400;
- O valor armazenado no endereço &400 é 7, que corresponde ao valor do elemento da lista;
- O valor armazenado no endereço &404 é 220, que corresponde ao endereço do próximo elemento da lista na memória.

O programa deve ser implementado utilizando o conjunto de instruções do processador RISC-V e deve ser executado no simulador RARS.

A função *main* a qual deve conter um menu com o acesso as seguintes funcionalidades

1) Inserir elemento na lista
   A função deve alocar memória e inserir o elemento de forma ordenada na lista.
2) Remover elemento da lista por indice
   A função deve remover o elemento de acordo com sua posição (indice) na lista. Considerar que os indice começa em 0 (zero).
3) Remover elemento da lista por valor
   A função deve remover da lista apenas o primeiro elemento encontrado com o valor recebido pela função.

4) Mostra todos os elementos da lista
Imprime na tela todos os elementos presentes na lista.

5) Mostra estatísticas
Imprime na tela as seguintes informações: quantidade de elementos presentes na lista; maior valor presente na lista; menor valor presente na lista; quantidade de inserções realizadas; quantidade de remoções realizadas;

6) Sair do programa
Encerra a execução do programa.

A função *main* deve ler do teclado os valores a serem passados como parâmetro para as funções. Por exemplo, para inserir um valor na lista, ainda dentro da função main deve-se ler o valor inteiro, que então será passado como parámetro para a função `insere_inteiro`.

Segue a descrição detalhada de cada uma das funções a serem implementadas pelo programa:

```
int insere_inteiro(int *head, int valor);
```

Parâmetros recebidos:
a0: a posição de memória do ponteiro para o inicio da lista;
a1: o valor a ser inserido;
Retorno da função:
a0: em caso de sucesso: 1;
em caso de falha: -1 (não foi possível inserir na lista);
a1: em caso de sucesso: o indice da posição inserida;
em caso de falha: -1;
Funcionalidade: a função deve alocar memória (usar ecall) e inserir o elemento de forma ordenada na lista;

```
int remove_por_indice(int *head, int indice);
```

Parâmetros recebidos:
a0: a posição de memória do ponteiro para o inicio da lista;
a1: o indice do elemento da lista a ser removido;
Retorno da função:
a0: em caso de sucesso: 1;
em caso de falha: -1 (não foi possível inserir na lista);
a1: em caso de sucesso: o valor presente na posição removida;
em caso de falha: -1;
Funcionalidade: a função deve retirar o elemento que esteja no indice informado;

```
int remove_por_valor(int *head, int valor);
```

Parâmetros recebidos:
a0: a posição de memória do ponteiro para o inicio da lista;
a1: o valor a ser removido;
Retorno da função:
a0: em caso de sucesso: 1;
em caso de falha: -1 (não foi possível inserir na lista);
a1: em caso de sucesso: o indice do elemento removido;
em caso de falha: -1;
em caso de falha: -1 caso não tenha sido possível remover da lista;
Funcionalidade: a função deve retirar o primeiro elemento com o valor informado presente nada lista;

```
void imprime_lista(int *head);
```

Parâmetros recebidos:

a0: a posição de memória do ponteiro para o inicio da lista;

Retorno da função: a função não possui retorno

Funcionalidade: a função deve mostrar na tela todos os elementos presentes na lista;

```
void estatistica();
```

Retorno da função: a função não possui retorno

Funcionalidade: mostrar as estatisticas apresentadas acima

Após o retorno da chamada de cada função, a função *main* deve mostrar na tela o resultado da operação. Por exemplo, no retorno da função insere_valor se o retorno for -1 deve mostrar uma mensagem como "Erro ao inserir elemento na lista" ou, em caso de sucesso "Valor X inserido na posição Z da lista".
Após informar o resultado da operação o programa deve mostrar novamente o menu principal para uma nova operação ser escolhida pelo usuário.

Detalhamento do Cronograma:

Entrega inicial (02/06) - lista de funcionalidades que devem estar prontas:

(a) a função *main* com o menu para chamada de todas as opções;
na função *main* a leitura do valor de entrada que será passado para a respectiva função para cada função;

(b) a chamada de cada uma das funções conforme a seleção no menu, passando os valores para a função;

(c) a mensagem informando o resultado da operação de cada uma das funções;

(d) a inserção de valores na lista (não precisa inserir de forma ordenada nesta entrega);

Para as atividades (b) e (c) não é necessário a implementação do corpo das funções, somente a chamada e o retorno das mesmas.

Entrega intermediária (12/06) - lista de funcionalidades que devem estar prontas:

(a) a função `insere_inteiro` completa;

(b) a função `imprime_lista` completa

(c) a função estatistica mostando o maior e o menor valor presentes na lista e a quantidade de inserções;

Entrega final (23/06) – Todas as funcionalidades da lista encadeada solicitadas

Nota do trabalho:

A nota será composta da seguinte forma:

- 50% implementação das funcionalidades
- 50% apresentação e explicações sobre a implementação

Penalidades das entregas inicial e intermediária do cronograma em relação a nota final do trabalho:

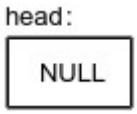
- não fez uma entrega: 10% por entrega não realizada;
- entrega atrasada: 5% por entrega atrasada;

- funcionalidade não implementada na entrega: 2% por cada funcionalidade não entregue.

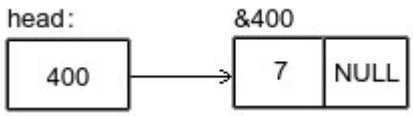
Em caso de plágio a nota será zero.

Exemplo:

Inicio



Após Inserir primeiro elemento



Próximos elementos: