UNIVERSIDADE DE BRASÍLIA
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO

116394 ORGANIZAÇÃO E ARQUITETURA DE COMPUTADORES

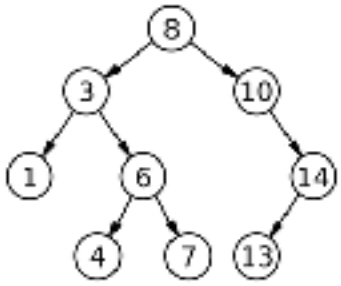
Trabalho III: Programação Assembler

OBJETIVO

Este trabalho objetiva o desenvolvimento de funções em *assembler* MIPS para realizar operações sobre árvores binárias de busca. As operações devem ser implementadas com algoritmos recursivos.

DESCRIÇÃO:

Árvores binárias de busca são árvores binárias onde os dados estão ordenados, de forma que todos os nós à esquerda de um nó raiz tem valores menores que este, e todos os nós à direita de um nó raiz tem valores maiores. Um exemplo de ABB é apresentado a seguir.



Um nó é representado no MARS por 3 palavras: (dado, esq, dir), onde ***dado*** é o valor contido no nó, ***esq*** e ***dir*** são os endereços dos nós à esquerda e à direita, respectivamente.

No MARS, uma árvore binária pode ser criada diretamente com *labels* indicando os endereços dos nós. Por exemplo, a árvore ilustrada acima poderia ser representada como segue.

```
.data
raiz:  .word 8  a0 a1
a0:    .word 3  a2 a3
a1:    .word 10 0  a4
a2:    .word 1  0  0
a3:    .word 6  a5 a6
a4:    .word 14 a7 0
a5:    .word 4  0  0
a6:    .word 7  0  0
a7:    .word 13 0  0
```

Um novo nó pode ser alocado reservando-se 3 palavras adjacentes na memória. O registrador ***$gp*** do MIPS localiza o início da área de dados estáticos na memória, que pode ser utilizado neste trabalho como *heap*, ou seja, área para alocação dinâmica de nós da árvore. Alocar um

nó utilizando o ponteiro *$gp* consiste em retornar o endereço atual de *$gp* e depois incrementá-lo de 12, para apontar para a próxima palavra livre na memória.

OPERAÇÕES

inserção recursiva:

```
void insere_rec(nodo raiz, int valor) {
  if (raiz == NULL) {
       raiz = malloc(sizeof(nodo));
       raiz->dado = valor;
       raiz->esq = NULL;
       raiz->dir = NULL;
       return raiz;
  }

  if (valor < raiz->dado)
    return insere_rec(raiz->esq, valor);

  if (valor > raiz->dado)
    return insere_rec(raiz->dir, valor);

  return raiz;
}
```

busca recursiva:

```
nodo busca_rec(nodo raiz, int valor) {
  if (raiz == NULL) return NULL;

  if (valor == raiz->dado)
    return raiz;

  if (valor > raiz->dado)
    return busca_rec(raiz->dir, valor);
  else
    return busca_rec(raiz->esq, valor);

}
```

De forma similar, implementar também a função ***size_rec()***, que retorna o número de nós da árvore.

ENTREGA

Entregar no Moodle em um arquivo compactado:

- breve descrição da implementação dos algoritmos em *assembler*
- descrição dos testes realizados
- código assembler