Universidade Federal da Bahia (UFBA)
Instituto de Matemática e Estatística (IME)
Departamento de Ciência da Computação (DCC)
MATA 40 - Estrutura de Dados e Algoritmos - Prof. George Lima
2019.1 - 30/05/2019 - Trabalho

Em algumas aplicações, necessita-se armazenar e fazer consulta a faixa de valores, representadas por intervalos. Por exemplo, em registros de ocorrências ou monitoramento de eventos, usualmente armazena-se informações indexadas pelo momento de início e fim dos acontecimentos. Uma das formas de suporte a estes tipos de aplicação é através da estrutura de dados conhecida como *árvore de intervalo*, que é uma árvore binária, com cada nó identificado por um par (a, b), representando um intervalo fechado $[a, b]$.

Para este trabalho, solicita-se uma implementação de uma árvore de intervalo, com funções para realizar as operações 1-5, descritas a seguir. Para executar cada uma das operações, o usuário deve escolher o número da operação e em seguida fornecer o parâmetro x. A implementação deve garantir que numa árvore contendo n nós, sua altura seja $O(\log n)$.

1. Incluir um intervalo x numa árvore de intervalo de raiz r.
```
void iTree_insert(interval *x, iTree *r);
```

2. Remover um intervalo x da árvore de intervalo de raíz r.
```
void iTree_delete(interval *x, iTree *r);
```

3. Buscar na árvore de raíz r um intervalo que tenha intersecção com o intervalo x. Caso não existam tais intervalos, retorna `NULL`.
```
interval *iTree_search(interval *x, iTree *r);
```

4. Listar todos os intervalos armazenados na árvore de raíx r que se sobrepõem ao intervalo x.
```
void iTree_overlap_list(interval *x, iTree *r);
```

5. Listar a árvore, imprimindo seus nós, um por nível. A árvore impressa deve estar visualmente compreensível, com cada nó-pai acima e à direita do seu filho esquerdo e acima e à esquerda do seu filho direito.
```
void iTree_display(iTree *r);
```

As seguintes definições serão necessárias:

```
typedef struct {
  int a,b;
} interval;

typedef struct {
  interval  v;
  int max;
  struct node *left, *right;
} iTree;
```