

[Árvore B+]

Gerado por Doxygen 1.9.3



<b>1 Índice das estruturas de dados</b>	<b>1</b>
1.1 Estruturas de dados	1
<b>2 Índice dos ficheiros</b>	<b>3</b>
2.1 Lista de ficheiros	3
<b>3 Documentação da classe</b>	<b>5</b>
3.1 Referência à estrutura artigo	5
3.1.1 Descrição detalhada	5
3.1.2 Documentação dos campos e atributos	5
3.1.2.1 ano	5
3.1.2.2 autor	6
3.1.2.3 DOI	6
3.1.2.4 id	6
3.1.2.5 invalido	6
3.1.2.6 palavraChave	6
3.1.2.7 revista	6
3.1.2.8 titulo	7
3.2 Referência à estrutura bm	7
3.2.1 Descrição detalhada	7
3.2.2 Documentação dos campos e atributos	7
3.2.2.1 felem	7
3.2.2.2 mgraui	7
3.2.2.3 raiz	8
3.3 Referência à estrutura bm_noh	8
3.3.1 Descrição detalhada	8
3.3.2 Documentação dos campos e atributos	8
3.3.2.1 chaves	8
3.3.2.2 dfilhos	8
3.3.2.3 eh_folha	9
3.3.2.4 filhos	9
3.3.2.5 mgraui	9
3.3.2.6 nchaves	9
3.3.2.7 pai	9
<b>4 Documentação do ficheiro</b>	<b>11</b>
4.1 Referência ao ficheiro src/bm.c	11
4.1.1 Documentação das funções	11
4.1.1.1 bm_escrutina()	11
4.1.1.2 bm_inic()	12
4.1.1.3 bm_inserir()	12
4.1.1.4 bm_inserir_nahfolha()	12
4.1.1.5 bm_lista()	13

---

4.1.1.6 bm_pesquisa()	13
4.2 bm.c	13
4.3 Referência ao ficheiro src/bm.h	15
4.3.1 Documentação dos tipos	15
4.3.1.1 bm	16
4.3.2 Documentação das funções	16
4.3.2.1 bm_escrutina()	16
4.3.2.2 bm_inic()	16
4.3.2.3 bm_insere()	16
4.3.2.4 bm_insere_nahfolha()	17
4.3.2.5 bm_lista()	17
4.3.2.6 bm_pesquisa()	18
4.4 bm.h	18
4.5 Referência ao ficheiro src/bm_noh.c	18
4.5.1 Documentação das funções	19
4.5.1.1 bm_noh_contem()	19
4.5.1.2 bm_noh_escrevedisc()	19
4.5.1.3 bm_noh_escrutina()	19
4.5.1.4 bm_noh_inic()	20
4.5.1.5 bm_noh_insere()	20
4.5.1.6 bm_noh_ledisc()	20
4.5.1.7 bm_noh_pesquisa()	21
4.5.1.8 bm_noh_pesquisa_folha()	21
4.5.1.9 bm_noh_split()	21
4.5.1.10 bm_noh_split_int()	22
4.5.1.11 padding()	22
4.6 bm_noh.c	22
4.7 Referência ao ficheiro src/bm_noh.h	26
4.7.1 Documentação das macros	27
4.7.1.1 MAXCHAVES	27
4.7.1.2 MAXFILHOS	27
4.7.1.3 NOME_ARQ	27
4.7.1.4 SUFIXO_BM	27
4.7.2 Documentação dos tipos	27
4.7.2.1 bm_noh	27
4.7.2.2 s_artigo	28
4.7.3 Documentação das funções	28
4.7.3.1 bm_noh_contem()	28
4.7.3.2 bm_noh_escrevedisc()	28
4.7.3.3 bm_noh_escrutina()	28
4.7.3.4 bm_noh_inic()	29
4.7.3.5 bm_noh_insere()	29

---

4.7.3.6 <code>bm_noh_ledisc()</code>	29
4.7.3.7 <code>bm_noh_pesquisa()</code>	30
4.7.3.8 <code>bm_noh_pesquisa_folha()</code>	30
4.7.3.9 <code>bm_noh_populaart()</code>	30
4.7.3.10 <code>bm_noh_split()</code>	31
4.7.3.11 <code>bm_noh_split_int()</code>	31
4.7.3.12 <code>padding()</code>	31
4.8 <code>bm_noh.h</code>	32
4.9 Referência ao ficheiro <code>src/checklist.c</code>	32
4.9.1 Documentação das funções	33
4.9.1.1 <code>tela_checklist()</code>	33
4.10 <code>checklist.c</code>	33
4.11 Referência ao ficheiro <code>src/checklist.h</code>	33
4.11.1 Documentação das macros	34
4.11.1.1 <code>ITEM_1</code>	34
4.11.1.2 <code>ITEM_2</code>	34
4.11.1.3 <code>ITEM_3</code>	34
4.11.1.4 <code>ITEM_4</code>	34
4.11.1.5 <code>ITEM_5</code>	35
4.11.1.6 <code>ITEM_6</code>	35
4.11.2 Documentação dos valores da enumeração	35
4.11.2.1 <code>anonymous enum</code>	35
4.11.3 Documentação das funções	35
4.11.3.1 <code>tela_checklist()</code>	35
4.12 <code>checklist.h</code>	36
4.13 Referência ao ficheiro <code>src/erro.h</code>	36
4.13.1 Documentação das funções	36
4.13.1.1 <code>erro()</code>	36
4.14 <code>erro.h</code>	37
4.15 Referência ao ficheiro <code>src/jcurses.h</code>	37
4.15.1 Documentação das macros	38
4.15.1.1 <code>CKLSITEM</code>	38
4.15.1.2 <code>CM</code>	38
4.15.1.3 <code>CMR</code>	38
4.15.1.4 <code>CMR1</code>	39
4.15.1.5 <code>INFO</code>	39
4.15.1.6 <code>INFO2</code>	39
4.15.1.7 <code>INFO_CARD</code>	39
4.15.1.8 <code>INFO_FILE</code>	39
4.15.1.9 <code>INFO_ST</code>	39
4.15.1.10 <code>LTELA</code>	40
4.15.1.11 <code>PNAPOS</code>	40

4.15.1.12 S_AZUL . . . . .	40
4.15.1.13 S_AZULB . . . . .	40
4.15.1.14 S_BRANCO . . . . .	40
4.15.1.15 S_BRANCOB . . . . .	40
4.15.1.16 S_CARD . . . . .	41
4.15.1.17 S_CHKLST . . . . .	41
4.15.1.18 S_CHKLST_DONE . . . . .	41
4.15.1.19 S_CINZA . . . . .	41
4.15.1.20 S_CINZAB . . . . .	41
4.15.1.21 S_CM . . . . .	41
4.15.1.22 S_CMR . . . . .	42
4.15.1.23 S_FILEN . . . . .	42
4.15.1.24 S_FORMAT . . . . .	42
4.15.1.25 S_INFO . . . . .	42
4.15.1.26 S_INFO2 . . . . .	42
4.15.1.27 S_INFO_ST . . . . .	42
4.15.1.28 S_INFO_ST_1 . . . . .	43
4.15.1.29 S_INV . . . . .	43
4.15.1.30 S_LIST . . . . .	43
4.15.1.31 S_LTELA . . . . .	43
4.15.1.32 S_MAGENTAB . . . . .	43
4.15.1.33 S_NADA . . . . .	43
4.15.1.34 S_NORM . . . . .	44
4.15.1.35 S_PISCA . . . . .	44
4.15.1.36 S_PISCARAP . . . . .	44
4.15.1.37 S_PRETO . . . . .	44
4.15.1.38 S_PRETOB . . . . .	44
4.15.1.39 S_TEST . . . . .	44
4.15.1.40 S_UNDERL . . . . .	45
4.15.1.41 S_VENN . . . . .	45
4.15.1.42 S_VERD . . . . .	45
4.15.1.43 S_VERDB . . . . .	45
4.15.1.44 S_VERM . . . . .	45
4.15.1.45 S_VERMB . . . . .	45
4.15.1.46 TIPO_DE_TERMINAL . . . . .	46
4.16 jcurses.h . . . . .	46
4.17 Referência ao ficheiro src/main.c . . . . .	46
4.17.1 Documentação das funções . . . . .	47
4.17.1.1 main() . . . . .	47
4.18 main.c . . . . .	47
4.19 Referência ao ficheiro src/testa.h . . . . .	49
4.20 testa.h . . . . .	49

---

---

4.21 Referência ao ficheiro src/testa_bm.c . . . . .	49
4.21.1 Documentação das funções . . . . .	49
4.21.1.1 tela_testa_bm() . . . . .	50
4.21.1.2 testa_arvore_bm() . . . . .	50
4.22 testa_bm.c . . . . .	50
4.23 Referência ao ficheiro src/testa_bm.h . . . . .	51
4.23.1 Documentação das funções . . . . .	51
4.23.1.1 tela_testa_bm() . . . . .	51
4.23.1.2 testa_arvore_bm() . . . . .	51
4.24 testa_bm.h . . . . .	51
<b>Índice</b>	<b>53</b>

---





# Capítulo 1

## Índice das estruturas de dados

### 1.1 Estruturas de dados

Lista das estruturas de dados com uma breve descrição:

<a href="#">artigo</a>	5
<a href="#">bm</a>	7
<a href="#">bm_noh</a>	8



## Capítulo 2

# Índice dos ficheiros

### 2.1 Lista de ficheiros

Lista de todos os ficheiros com uma breve descrição:

src/ <a href="#">bm.c</a>	11
src/ <a href="#">bm.h</a>	15
src/ <a href="#">bm_noh.c</a>	18
src/ <a href="#">bm_noh.h</a>	26
src/ <a href="#">checklist.c</a>	32
src/ <a href="#">checklist.h</a>	33
src/ <a href="#">erro.h</a>	36
src/ <a href="#">jcurses.h</a>	37
src/ <a href="#">main.c</a>	46
src/ <a href="#">testa.h</a>	49
src/ <a href="#">testa_bm.c</a>	49
src/ <a href="#">testa_bm.h</a>	51



## Capítulo 3

# Documentação da classe

### 3.1 Referência à estrutura artigo

```
#include <bm_noh.h>
```

#### Campos de Dados

- int [id](#)
- int [ano](#)
- char [autor](#) [200]
- char [titulo](#) [200]
- char [revista](#) [200]
- char [DOI](#) [20]
- char [palavraChave](#) [200]
- char [invalido](#)

#### 3.1.1 Descrição detalhada

Definido na linha [14](#) do ficheiro [bm\\_noh.h](#).

#### 3.1.2 Documentação dos campos e atributos

##### 3.1.2.1 ano

```
int ano
```

Definido na linha [17](#) do ficheiro [bm\\_noh.h](#).

---

#### 3.1.2.2 autor

```
char autor[200]
```

Definido na linha 18 do ficheiro [bm\\_noh.h](#).

#### 3.1.2.3 DOI

```
char DOI[20]
```

Definido na linha 21 do ficheiro [bm\\_noh.h](#).

#### 3.1.2.4 id

```
int id
```

Definido na linha 16 do ficheiro [bm\\_noh.h](#).

#### 3.1.2.5 invalido

```
char invalido
```

Definido na linha 23 do ficheiro [bm\\_noh.h](#).

#### 3.1.2.6 palavraChave

```
char palavraChave[200]
```

Definido na linha 22 do ficheiro [bm\\_noh.h](#).

#### 3.1.2.7 revista

```
char revista[200]
```

Definido na linha 20 do ficheiro [bm\\_noh.h](#).

---

### 3.1.2.8 titulo

```
char titulo[200]
```

Definido na linha 19 do ficheiro [bm\\_noh.h](#).

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- [src/bm\\_noh.h](#)

## 3.2 Referência à estrutura bm

```
#include <bm.h>
```

### Campos de Dados

- struct [bm\\_noh](#) \* [raiz](#)
- int [mgrau](#)
- int \* [felem](#)

### 3.2.1 Descrição detalhada

Definido na linha 11 do ficheiro [bm.h](#).

### 3.2.2 Documentação dos campos e atributos

#### 3.2.2.1 felem

```
int * felem
```

Definido na linha 14 do ficheiro [bm.h](#).

#### 3.2.2.2 mgrau

```
int mgrau
```

Definido na linha 14 do ficheiro [bm.h](#).

---

### 3.2.2.3 raiz

```
struct bm\_noh* raiz
```

Definido na linha [13](#) do ficheiro [bm.h](#).

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- [src/bm.h](#)

## 3.3 Referência à estrutura [bm\\_noh](#)

```
#include <bm\_noh.h>
```

### Campos de Dados

- int \* [chaves](#)
- int [nchaves](#)
- int [mgrau](#)
- struct [bm\\_noh](#) \*\* [filhos](#)
- struct [bm\\_noh](#) \* [pai](#)
- long \*\* [dfilhos](#)
- char [eh\\_folha](#)

### 3.3.1 Descrição detalhada

Definido na linha [26](#) do ficheiro [bm\\_noh.h](#).

### 3.3.2 Documentação dos campos e atributos

#### 3.3.2.1 chaves

```
int* chaves
```

Definido na linha [28](#) do ficheiro [bm\\_noh.h](#).

#### 3.3.2.2 dfilhos

```
long** dfilhos
```

Definido na linha [32](#) do ficheiro [bm\\_noh.h](#).

---



### 3.3.2.3 `eh_folha`

```
char eh_folha
```

Definido na linha 33 do ficheiro `bm_noh.h`.

### 3.3.2.4 `filhos`

```
struct bm_noh** filhos
```

Definido na linha 30 do ficheiro `bm_noh.h`.

### 3.3.2.5 `mgrau`

```
int mgrau
```

Definido na linha 29 do ficheiro `bm_noh.h`.

### 3.3.2.6 `nchaves`

```
int nchaves
```

Definido na linha 28 do ficheiro `bm_noh.h`.

### 3.3.2.7 `pai`

```
struct bm_noh* pai
```

Definido na linha 31 do ficheiro `bm_noh.h`.

A documentação para esta estrutura foi gerada a partir do seguinte ficheiro:

- `src/bm_noh.h`



## Capítulo 4

# Documentação do ficheiro

### 4.1 Referência ao ficheiro src/bm.c

```
#include "bm.h"
#include <stdio.h>
#include <stdlib.h>
```

#### Funções

- `bm * bm_inic` (int i)
- void `bm_escrutina` (`bm *b`)
- `bm_noh * bm_pesquisa` (`bm *b`, int chave)
- void `bm_lista` (`bm_noh *aux`, int chave)
- `bm_noh * bm_insere_nahfolha` (`bm_noh *raiz`, int chave, `s_artigo *art`)
- void `bm_insere` (`bm *b`, int chave, `s_artigo *art`)

#### 4.1.1 Documentação das funções

##### 4.1.1.1 `bm_escrutina()`

```
void bm_escrutina (
    bm * b )
```

faz a impressão da árvore b+

#### Parâmetros

<code>bm</code>	b árvore
-----------------	----------

Definido na linha [28](#) do ficheiro `bm.c`.

---

#### 4.1.1.2 `bm_inic()`

```
bm * bm_inic (
    int i )
```

faz a inicialização da árvore b+

##### Parâmetros

<i>int</i>	i ordem da árvore
------------	-------------------

Definido na linha 15 do ficheiro `bm.c`.

#### 4.1.1.3 `bm_inserere()`

```
void bm_inserere (
    bm * b,
    int chave,
    s_artigo * art )
```

faz a inserção de uma chave na árvore b+

##### Parâmetros

<i>bm</i>	b árvore
<i>int</i>	chave a ser buscada
<i>s_artigo</i>	art artigo a ser armazenado

se árvore vazia

se árvore nchaves vazia

aloca memória para nó raiz

atualiza o número de chaves

insere chave

Definido na linha 138 do ficheiro `bm.c`.

#### 4.1.1.4 `bm_inserere_nahfolha()`

```
bm_noh * bm_inserere_nahfolha (
    bm_noh * raiz,
    int chave,
    s_artigo * art )
```

faz a inserção de uma chave na folha da árvore b+

---

## Parâmetros

<i>bm</i>	b arvore
<i>int</i>	chave a ser buscada
<i>s_artigo</i>	art artigo a ser armazenado

Definido na linha 85 do ficheiro [bm.c](#).

#### 4.1.1.5 bm\_lista()

```
void bm_lista (
    bm_noh * aux,
    int chave )
```

Definido na linha 57 do ficheiro [bm.c](#).

#### 4.1.1.6 bm\_pesquisa()

```
bm_noh * bm_pesquisa (
    bm * b,
    int chave )
```

faz a pesquisa de uma chave na arvore b+

## Parâmetros

<i>bm</i>	b arvore
<i>int</i>	chave a ser buscada

Definido na linha 39 do ficheiro [bm.c](#).

## 4.2 bm.c

[Ir para a documentação deste ficheiro.](#)

```
00001 /*
00002  * Grupo 11
00003  * alunos: joilnen leite, daniel morais, matheus silva
00004  *
00005  *
00006  */
00007 #include "bm.h"
00008 #include <stdio.h>
00009 #include <stdlib.h>
00010
00015 bm *bm_inic(int i)
00016 {
00017     bm *b = (bm *) malloc(sizeof(bm));
00018     b->raiz = NULL;
00019     b->mgrau = i;
00020 }
```

```

00021     return b;
00022 }
00023
00028 void bm_escrutina(bm *b)
00029 {
00030     if (b->raiz)
00031         bm_noh_escrutina(b->raiz, 0);
00032 }
00033
00039 bm_noh *bm_pesquisa(bm *b, int chave)
00040 {
00041     return (b->raiz) ? bm_noh_pesquisa(b->raiz, chave) : NULL;
00042 }
00043
00044 static void bm_grava_art(bm_noh *raiz, int chave, s_artigo *art)
00045 {
00046     int i;
00047     for (i = 0; i < raiz->nchaves; i++)
00048     {
00049         if (raiz->chaves[i] == chave)
00050         {
00051             art->id = chave;
00052             bm_noh_escrevedisc(raiz, art);
00053         }
00054     }
00055 }
00056
00057 void bm_lista(bm_noh *aux, int chave){
00058     int i;
00059
00060     if (!aux)
00061         return;
00062
00063     if (aux->eh_folha)
00064     {
00065         printf("->");
00066         for(int j = 0; j < aux->nchaves; j++)
00067             printf(" %d ", aux->chaves[j]);
00068     }
00069     else
00070     {
00071         for(i = 0; i <= aux->nchaves; i++)
00072         {
00073             if (aux->chaves[i] >= chave)
00074                 bm_lista(aux->filhos[i+1], chave);
00075         }
00076     }
00077 }
00078
00085 bm_noh *bm_insere_nahfolha(bm_noh *raiz, int chave, s_artigo *art)
00086 {
00087     int i;
00088     if (!raiz->eh_folha)
00089     {
00090         for (i = raiz->nchaves - 1; i >= 0; --i)
00091         {
00092             if (raiz->chaves[i] < chave)
00093             {
00094                 bm_insere_nahfolha(raiz->filhos[i + 1], chave, art);
00095                 i = -1;
00096             }
00097         }
00098     }
00099     else
00100     {
00101         bm_noh_insere(raiz, chave);
00102         if (raiz->nchaves == MAXCHAVES(raiz->mgrau) + 1)
00103         {
00104             if (!raiz->pai)
00105             {
00106                 bm_noh *aux = bm_noh_inic(raiz->mgrau, 0);
00107                 aux->filhos[0] = raiz;
00108                 raiz->pai = aux;
00109                 i = 0;
00110             }
00111             else
00112             {
00113                 i = raiz->pai->nchaves;
00114                 while (i >= 0 && raiz->pai->chaves[i] > chave)
00115                 {
00116                     raiz->chaves[i + 1] = raiz->chaves[i];
00117                     --i;
00118                 }
00119             }
00120             bm_noh_split(raiz->pai, raiz, i);
00121         }
00122     }

```

```

00123         bm_grava_art(raiz, chave, art);
00124     }
00125
00126     while (raiz->pai)
00127         raiz = raiz->pai;
00128
00129     return raiz;
00130 }
00131
00132 void bm_insere(bm *b, int chave, s_artigo *art)
00133 {
00134     if (b->raiz)
00135     {
00136         if (bm_noh_pesquisa_folha(b->raiz, chave))
00137         {
00138             printf("\nChave ja inserida na arvore\n");
00139             return;
00140         }
00141         //printf("\n-----\nO QUE ENTOU:");
00142         //bm_escrutina(b);
00143         b->raiz = bm_insere_nahfolha(b->raiz, chave, art);
00144         //printf("\nO QUE SAIU:\n");
00145         //bm_escrutina(b);
00146     }
00147     else
00148     {
00149         //printf("\n INICIA COM %d\n", chave);
00150         b->raiz = bm_noh_inic(b->mgrau, 1);
00151         b->raiz->nchaves = 1;
00152         b->raiz->chaves[0] = chave;
00153         bm_grava_art(b->raiz, chave, art);
00154     }
00155 }
00156
00157
00158
00159
00160
00161
00162
00163
00164
00165
00166

```

## 4.3 Referência ao ficheiro src/bm.h

```
#include "bm_noh.h"
```

### Estruturas de Dados

- struct `bm`

### Definições de tipos

- typedef struct `bm` `bm`

### Funções

- `bm * bm_inic` (int i)
- void `bm_escrutina` (bm \*b)
- `bm_noh * bm_pesquisa` (bm \*b, int chave)
- void `bm_insere` (bm \*b, int chave, s\_artigo \*art)
- `bm_noh * bm_insere_nahfolha` (bm\_noh \*raiz, int chave, s\_artigo \*art)
- void `bm_lista` (bm\_noh \*aux, int chave)

#### 4.3.1 Documentação dos tipos

#### 4.3.1.1 `bm`

```
typedef struct bm bm
```

### 4.3.2 Documentação das funções

#### 4.3.2.1 `bm_escrutina()`

```
void bm_escrutina (  
    bm * b )
```

faz a impressão da árvore b+

Parâmetros

<i>bm</i>	b árvore
-----------	----------

Definido na linha 28 do ficheiro `bm.c`.

#### 4.3.2.2 `bm_inic()`

```
bm * bm_inic (  
    int i )
```

faz a inicialização da árvore b+

Parâmetros

<i>int</i>	i ordem da árvore
------------	-------------------

Definido na linha 15 do ficheiro `bm.c`.

#### 4.3.2.3 `bm_insere()`

```
void bm_insere (  
    bm * b,  
    int chave,  
    s_artigo * art )
```

faz a inserção de uma chave na árvore b+

---



## Parâmetros

<i>bm</i>	b árvore
<i>int</i>	chave a ser buscada
<i>s_artigo</i>	art artigo a ser armazenado

se árvore vazia

se árvore nchaves vazia

aloca memória para noh raiz

atualiza o número de chaves

insere chave

Definido na linha 138 do ficheiro [bm.c](#).

#### 4.3.2.4 **bm\_insere\_nahfolha()**

```
bm_noh * bm_insere_nahfolha (  
    bm_noh * raiz,  
    int chave,  
    s_artigo * art )
```

faz a inserção de uma chave na folha da árvore b+

## Parâmetros

<i>bm</i>	b árvore
<i>int</i>	chave a ser buscada
<i>s_artigo</i>	art artigo a ser armazenado

Definido na linha 85 do ficheiro [bm.c](#).

#### 4.3.2.5 **bm\_lista()**

```
void bm_lista (  
    bm_noh * aux,  
    int chave )
```

Definido na linha 57 do ficheiro [bm.c](#).

---

#### 4.3.2.6 `bm_pesquisa()`

```
bm_noh * bm_pesquisa (
    bm * b,
    int chave )
```

faz a pesquisa de uma chave na arvore b+

##### Parâmetros

<i>bm</i>	b arvore
<i>int</i>	chave a ser buscada

Definido na linha 39 do ficheiro `bm.c`.

## 4.4 `bm.h`

[Ir para a documentação deste ficheiro.](#)

```
00001 /*
00002  * alunos: joilnen leite, daniel morais, matheus silva
00003  *
00004  *
00005  */
00006 #ifndef BM_
00007 #define BM_
00008
00009 #include "bm_noh.h"
00010
00011 typedef struct bm
00012 {
00013     struct bm_noh *raiz;
00014     int mgraui, *felem;
00015 } bm;
00016
00017 bm *bm_inic(int i);
00018 void bm_escrutina(bm *b);
00019 bm_noh *bm_pesquisa(bm *b, int chave);
00020 void bm_insere(bm *b, int chave, s_artigo *art);
00021 bm_noh *bm_insere_nahfolha(bm_noh *raiz, int chave, s_artigo *art);
00022 void bm_lista(bm_noh *aux, int chave);
00023
00024 #endif
```

## 4.5 Referência ao ficheiro `src/bm_noh.c`

```
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
#include "bm_noh.h"
#include "erro.h"
```

### Funções

- `bm_noh * bm_noh_inic` (int i, char eh\_folha)
- void `bm_noh_split` (bm\_noh \*bmn, bm\_noh \*y, int i)
- `bm_noh * bm_noh_split_int` (bm\_noh \*bmn, bm\_noh \*y, bm\_noh \*z)

- void `bm_noh_insere` (`bm_noh *bmn`, int chave)
- void `padding` (char ch, int n)
- void `bm_noh_escrutina` (`bm_noh *bmn`, int nivel)
- `bm_noh * bm_noh_pesquisa_folha` (`bm_noh *bmn`, int chave)
- `bm_noh * bm_noh_pesquisa` (`bm_noh *bmn`, int chave)
- int `bm_noh_contem` (`bm_noh *bmn`, int chave)
- void `bm_noh_escrevedisc` (`bm_noh *bmn`, `s_artigo *art`)
- `s_artigo * bm_noh_ledisc` (`bm_noh *bmn`, int chave)

## 4.5.1 Documentação das funções

### 4.5.1.1 `bm_noh_contem()`

```
int bm_noh_contem (  
    bm_noh * bmn,  
    int chave )
```

Definido na linha 248 do ficheiro `bm_noh.c`.

### 4.5.1.2 `bm_noh_escrevedisc()`

```
void bm_noh_escrevedisc (  
    bm_noh * bmn,  
    s_artigo * art )
```

escreve conteúdo da página folha no disco

Parâmetros

<code><i>bm_noh</i></code>	bmn página folha
----------------------------	------------------

Definido na linha 260 do ficheiro `bm_noh.c`.

### 4.5.1.3 `bm_noh_escrutina()`

```
void bm_noh_escrutina (  
    bm_noh * bmn,  
    int nivel )
```

faz a impressão das chaves na página

---

**Parâmetros**

<i>bm_noh</i>	bm_n página raiz
<i>int</i>	nível nível da árvore

Definido na linha 185 do ficheiro [bm\\_noh.c](#).

**4.5.1.4 bm\_noh\_inic()**

```
bm_noh * bm_noh_inic (
    int i,
    char eh_folha )
```

faz a inicialização de um nó

**Parâmetros**

<i>int</i>	i grau da página
<i>char</i>	eh_folha flag para saber se a página é interna ou externa

Definido na linha 19 do ficheiro [bm\\_noh.c](#).

**4.5.1.5 bm\_noh\_insere()**

```
void bm_noh_insere (
    bm_noh * bmn,
    int chave )
```

faz a insecção de uma chave na página

**Parâmetros**

<i>bm_noh</i>	bm_n página
<i>int</i>	chave chave a ser inserida

```
for (j = 0; j < bmn->nchaves; j++) printf("%d ", bmn->chaves[j]); printf("\n");
```

Definido na linha 139 do ficheiro [bm\\_noh.c](#).

**4.5.1.6 bm\_noh\_ledisc()**

```
s_artigo * bm_noh_ledisc (
    bm_noh * bmn,
    int chave )
```

ler conteúdo do disco

#### Parâmetros

<i>bm_noh</i>	bm_n página
<i>int</i>	chave chave para a identificação no disco

Definido na linha 311 do ficheiro [bm\\_noh.c](#).

#### 4.5.1.7 **bm\_noh\_pesquisa()**

```
bm_noh * bm_noh_pesquisa (
    bm_noh * bmn,
    int chave )
```

pesquisa uma chave na árvore

#### Parâmetros

<i>bm_noh</i>	bm_n página raiz
<i>int</i>	chave chave a ser buscada

Definido na linha 236 do ficheiro [bm\\_noh.c](#).

#### 4.5.1.8 **bm\_noh\_pesquisa\_folha()**

```
bm_noh * bm_noh_pesquisa_folha (
    bm_noh * bmn,
    int chave )
```

pesquisa uma chave na árvore

#### Parâmetros

<i>bm_noh</i>	bm_n página raiz
<i>int</i>	chave chave a ser buscada

Definido na linha 217 do ficheiro [bm\\_noh.c](#).

#### 4.5.1.9 **bm\_noh\_split()**

```
void bm_noh_split (
    bm_noh * bmn,
```

```
bm_noh * y,  
int i )
```

realiza a partição de uma página folha

#### Parâmetros

<i>bm_noh</i>	bmn
<i>bm_noh</i>	y
<i>int</i>	i

Definido na linha [39](#) do ficheiro [bm\\_noh.c](#).

#### 4.5.1.10 bm\_noh\_split\_int()

```
bm_noh * bm_noh_split_int (  
    bm_noh * bmn,  
    bm_noh * y,  
    bm_noh * z )
```

realiza a partição de uma página interna

#### Parâmetros

<i>bm_noh</i>	bmn
<i>bm_noh</i>	y
<i>int</i>	i

Definido na linha [78](#) do ficheiro [bm\\_noh.c](#).

#### 4.5.1.11 padding()

```
void padding (  
    char ch,  
    int n )
```

faz a impressão de espaços para a função escrutina

Definido na linha [173](#) do ficheiro [bm\\_noh.c](#).

## 4.6 bm\_noh.c

[Ir para a documentação deste ficheiro.](#)

```
00001 /* Grupo 11  
00002 * alunos: joilnen leite, daniel morais, matheus silva
```

---

```

00003  *
00004  *
00005  */
00006 #include <stdio.h>
00007 #include <stdlib.h>
00008 #include <string.h>
00009
00010 #include "bm_noh.h"
00011 #include "erro.h"
00012
00019 bm_noh *bm_noh_inic(int i, char eh_folha)
00020 {
00021     bm_noh *n = (bm_noh *)malloc(sizeof(bm_noh));
00022     n->mgrau = i;
00023     n->chaves = (int *)malloc(sizeof(int) * MAXCHAVES(n->mgrau));
00024     n->filhos = (bm_noh **)malloc(sizeof(bm_noh) * MAXFILHOS(n->mgrau));
00025     n->dfilhos = (long **)malloc(sizeof(long) * MAXCHAVES(n->mgrau));
00026     n->pai = NULL;
00027
00028     n->nchaves = 0;
00029     n->eh_folha = eh_folha;
00030     return n;
00031 }
00032
00039 void bm_noh_split(bm_noh *bmn, bm_noh *y, int i)
00040 {
00041     int j;
00042     printf("\nbm_noh_split\n");
00043     bm_noh *z = bm_noh_inic(y->mgrau, y->eh_folha);
00044     z->pai = bmn;
00045     y->pai = bmn;
00046     //Como z é sempre um noh folha, ele tambem recebe os ponteiros de arquivo de y
00047     for (j = 0; j < bmn->mgrau; j++)
00048     {
00049         z->chaves[j] = y->chaves[j + bmn->mgrau];
00050         z->>dfilhos[j] = y->dfilhos[j + bmn->mgrau];
00051         z->nchaves++;
00052     }
00053     y->nchaves -= z->nchaves;
00054
00055     for (j = bmn->nchaves; j >= i + 1; j--) {
00056         bmn->filhos[j + 1] = bmn->filhos[j];
00057         bmn->>dfilhos[j] = bmn->>dfilhos[j-1];
00058     }
00059     bmn->filhos[i + 1] = z;
00060     for (j = bmn->nchaves - 1; j >= i; j--)
00061         bmn->chaves[j + 1] = bmn->chaves[j];
00062
00063     for (j = bmn->nchaves - 1; j >= i; j--)
00064         bmn->chaves[j + 1] = bmn->chaves[j];
00065
00066     bmn->chaves[i] = y->chaves[bmn->mgrau];
00067     bmn->nchaves++;
00068     bmn = bm_noh_split_int(bmn, y, z);
00069 }
00070
00071
00078 bm_noh *bm_noh_split_int(bm_noh *bmn, bm_noh *y, bm_noh *z)
00079 {
00080     printf("\nbm_noh_split_int\n");
00081     int j;
00082     if (bmn->nchaves == MAXCHAVES(bmn->mgrau) + 1)
00083     {
00084         bm_noh *w = bm_noh_inic(bmn->mgrau, 0);
00085         for (j = 1; j < bmn->mgrau; j++)
00086         {
00087             w->chaves[j - 1] = bmn->chaves[j + bmn->mgrau];
00088             w->dfilhos[j - 1] = bmn->dfilhos[j + bmn->mgrau];
00089             w->filhos[j - 1] = bmn->filhos[j + bmn->mgrau];
00090             w->filhos[j] = bmn->filhos[j + bmn->mgrau + 1];
00091             w->nchaves++;
00092             bmn->nchaves--;
00093         }
00094         bmn->nchaves--;
00095
00096         bm_noh *noh_novo;
00097         if (!bmn->pai)
00098         {
00099             noh_novo = bm_noh_inic(bmn->mgrau, 0);
00100             bmn->pai = noh_novo;
00101             noh_novo->chaves[0] = bmn->chaves[bmn->mgrau];
00102             noh_novo->filhos[0] = bmn;
00103             noh_novo->filhos[1] = w;
00104         }
00105         else
00106         {
00107             noh_novo = bmn->pai;

```

```

00108         for (j = 1; j <= noh_novo->nchaves; j++)
00109         {
00110             if (noh_novo->chaves[j] == 0)
00111             {
00112                 noh_novo->chaves[j] = bmn->chaves[bmn->mgrau];
00113                 noh_novo->filhos[j] = bmn;
00114                 noh_novo->filhos[j + 1] = w;
00115             }
00116         }
00117     }
00118     noh_novo->nchaves++;
00119
00120     w->pai = noh_novo;
00121     y->pai = w;
00122     z->pai = w;
00123
00124     bm_noh_split_int(noh_novo, bmn, w);
00125 }
00126 else
00127 {
00128     return bmn;
00129 }
00130 }
00131
00139 void bm_noh_insere(bm_noh *bmn, int chave)
00140 {
00141     int i = bmn->nchaves - 1, j;
00142     if (bmn->eh_folha)
00143     {
00144         while (i >= 0 && bmn->chaves[i] > chave)
00145         {
00146             bmn->chaves[i + 1] = bmn->chaves[i];
00147             --i;
00148         }
00149         bmn->chaves[i + 1] = chave;
00150         bmn->nchaves++;
00151     }
00152     else
00153     {
00154         while (i >= 0 && bmn->chaves[i] > chave)
00155             --i;
00156         bm_noh_insere(bmn->filhos[i + 1], chave);
00157         if (bmn->filhos[i + 1]->nchaves == MAXCHAVES (bmn->mgrau) + 1)
00158         {
00159             bm_noh_split(bmn, bmn->filhos[i + 1], i + 1);
00160             i = bmn->nchaves - 1;
00161         }
00162     }
00163 }
00164 }
00165 }
00166 }
00167 }
00168
00173 void padding(char ch, int n)
00174 {
00175     for (int i = 0; i < n; i++)
00176         putchar(ch);
00177 }
00178
00185 void bm_noh_escrutina(bm_noh *bmn, int nivel)
00186 {
00187     int i;
00188     if (bmn->eh_folha)
00189     {
00190         for (i = 0; i < bmn->nchaves; i++)
00191         {
00192             printf("\n");
00193             padding('\t', nivel + 1);
00194             printf("%d ", bmn->chaves[i]);
00195         }
00196     }
00197     else
00198     {
00199         bm_noh_escrutina(bmn->filhos[0], nivel + 1);
00200         for (i = 0; i < bmn->nchaves; i++)
00201         {
00202             printf("\n");
00203             padding('\t', nivel);
00204             printf("%d ", bmn->chaves[i]);
00205             //printf("%d[%d] ", bmn->chaves[i], nivel);
00206             bm_noh_escrutina(bmn->filhos[i + 1], nivel + 1);
00207         }
00208     }
00209 }
00210
00217 bm_noh *bm_noh_pesquisa_folha(bm_noh *bmn, int chave)
00218 {
00219     int i = 0;
00220     while (i < bmn->nchaves && chave > bmn->chaves[i])
00221         i++;

```



```

00222     if (bmn->chaves[i] == chave && bmn->eh_folha)
00223         return bmn;
00224     else if (bmn->eh_folha)
00225         return NULL;
00226
00227     return bm_noh_pesquisa(bmn->filhos[i], chave);
00228 }
00229
00236 bm_noh *bm_noh_pesquisa(bm_noh *bmn, int chave)
00237 {
00238     int i = 0;
00239     while (i < bmn->nchaves && chave > bmn->chaves[i])
00240         i++;
00241     if (bmn->chaves[i] == chave)
00242         return bmn;
00243     if (bmn->eh_folha)
00244         return NULL;
00245     return bm_noh_pesquisa(bmn->filhos[i], chave);
00246 }
00247
00248 int bm_noh_contem(bm_noh *bmn, int chave)
00249 {
00250     if (bm_noh_pesquisa(bmn, chave))
00251         return 1;
00252     return 0;
00253 }
00254
00260 void bm_noh_escrevedisc(bm_noh *bmn, s_artigo *art)
00261 {
00262     FILE *parq;
00263     int j;
00264     int quantos_arts = 0;
00265
00266     char *tmp = NULL;
00267
00268     if (!bmn->eh_folha)
00269         erro(__func__, "arquivo: foi tentado escrever em disco pagina nao folha");
00270
00271     tmp = (char *)malloc(sizeof(char) * (strlen(NOME_ARQ) + strlen(SUFIXO_BM) + 1));
00272     strcpy(tmp, NOME_ARQ"SUFIXO_BM");
00273
00274
00275     if (!(parq = fopen(tmp, "w+")))
00276     {
00277         free(tmp);
00278         erro(__func__, "arquivo: erro na criacao do arquivo .bm");
00279     }
00280     free(tmp);
00281
00282     fseek(parq, 0, SEEK_END);
00283     if (ftell(parq) > sizeof(int))
00284     {
00285         rewind(parq);
00286         fread(&quantos_arts, sizeof(int), 1, parq);
00287     }
00288     fseek(parq, 0, SEEK_END);
00289     quantos_arts++;
00290     fwrite(&quantos_arts, sizeof(int), 1, parq);
00291     fseek(parq, 0, SEEK_END);
00292
00293     for (j = 0; bmn->eh_folha && j < bmn->nchaves; j++)
00294     {
00295         if (bmn->chaves[j] == art->id)
00296         {
00297             bmn->dfilhos[j] = (long *)ftell(parq);
00298             fwrite(art, sizeof(s_artigo), 1, parq);
00299             break;
00300         }
00301     }
00302     fclose(parq);
00303 }
00304
00311 s_artigo *bm_noh_ledisc(bm_noh *bmn, int chave)
00312 {
00313     FILE *parq;
00314     int j;
00315
00316     char *tmp;
00317
00318     if (!bmn->eh_folha)
00319         erro(__func__, "arquivo: foi tentado ler em disco pagina nao folha");
00320
00321     tmp =
00322     (char *)malloc(sizeof(char) * (strlen(NOME_ARQ) + strlen(SUFIXO_BM) + 1));
00323     strcpy(tmp, NOME_ARQ"SUFIXO_BM");
00324
00325     if (!(parq = fopen(tmp, "r")))

```

```

00326     {
00327         free(tmp);
00328         erro(__func__, "arquivo: erro na leitura do arquivo .bm");
00329     }
00330     free(tmp);
00331
00332     for (j = 0; bmn->eh_folha && j < bmn->nchaves; j++)
00333     {
00334         s_artigo *a = (s_artigo *)malloc(sizeof(s_artigo));
00335         if (chave == bmn->chaves[j])
00336         {
00337             rewind(parq);
00338             fseek(parq, *bmn->dfilhos[j], SEEK_SET);
00339             fread(a, sizeof(s_artigo), 1, parq);
00340             fclose(parq);
00341             return a;
00342         }
00343     }
00344     free(a);
00345 }
00346 fclose(parq);
00347
00348 return NULL;
00349 }
00350
00351

```

## 4.7 Referência ao ficheiro src/bm\_noh.h

### Estruturas de Dados

- struct [artigo](#)
- struct [bm\\_noh](#)

### Macros

- #define [NOME\\_ARQ](#) "bmais\_arquivo"
- #define [SUFIXO\\_BM](#) ".bm"
- #define [MAXFILHOS](#)(m) 2 \* m
- #define [MAXCHAVES](#)(m) [MAXFILHOS](#)(m) - 1

### Definições de tipos

- typedef struct [artigo](#) [s\\_artigo](#)
- typedef struct [bm\\_noh](#) [bm\\_noh](#)

### Funções

- [bm\\_noh](#) \* [bm\\_noh\\_inic](#) (int i, char eh\_folha)
- void [bm\\_noh\\_split](#) ([bm\\_noh](#) \*bmn, [bm\\_noh](#) \*y, int i)
- [bm\\_noh](#) \* [bm\\_noh\\_split\\_int](#) ([bm\\_noh](#) \*bmn, [bm\\_noh](#) \*y, [bm\\_noh](#) \*z)
- void [bm\\_noh\\_insere](#) ([bm\\_noh](#) \*bmn, int chave)
- void [padding](#) (char ch, int n)
- void [bm\\_noh\\_escrutina](#) ([bm\\_noh](#) \*bmn, int nivel)
- [bm\\_noh](#) \* [bm\\_noh\\_pesquisa](#) ([bm\\_noh](#) \*bmn, int k)
- [bm\\_noh](#) \* [bm\\_noh\\_pesquisa\\_folha](#) ([bm\\_noh](#) \*bmn, int chave)
- int [bm\\_noh\\_contem](#) ([bm\\_noh](#) \*bmn, int chave)
- void [bm\\_noh\\_escrevedisc](#) ([bm\\_noh](#) \*bmn, [s\\_artigo](#) \*art)
- [s\\_artigo](#) \* [bm\\_noh\\_ledisc](#) ([bm\\_noh](#) \*bmn, int chave)
- void [bm\\_noh\\_populaart](#) (int chave, [s\\_artigo](#) \*art)

## 4.7.1 Documentação das macros

### 4.7.1.1 MAXCHAVES

```
#define MAXCHAVES(  
    m ) MAXFILHOS(m) - 1
```

Definido na linha 58 do ficheiro `bm_noh.h`.

### 4.7.1.2 MAXFILHOS

```
#define MAXFILHOS(  
    m ) 2 * m
```

Definido na linha 57 do ficheiro `bm_noh.h`.

### 4.7.1.3 NOME\_ARQ

```
#define NOME_ARQ "bmais_arquivo"
```

nome padrao do arquivo

Definido na linha 10 do ficheiro `bm_noh.h`.

### 4.7.1.4 SUFIXO\_BM

```
#define SUFIXO_BM ".bm"
```

sufixos utilizado no aquivo de dados e de indices

Definido na linha 12 do ficheiro `bm_noh.h`.

## 4.7.2 Documentação dos tipos

### 4.7.2.1 bm\_noh

```
typedef struct bm_noh bm_noh
```

---

#### 4.7.2.2 s\_artigo

```
typedef struct artigo s_artigo
```

### 4.7.3 Documentação das funções

#### 4.7.3.1 bm\_noh\_contem()

```
int bm_noh_contem (
    bm_noh * bmn,
    int chave )
```

Definido na linha 248 do ficheiro [bm\\_noh.c](#).

#### 4.7.3.2 bm\_noh\_escrevedisc()

```
void bm_noh_escrevedisc (
    bm_noh * bmn,
    s_artigo * art )
```

escreve conteúdo da página folha no disco

##### Parâmetros

<a href="#">bm_noh</a>	bmn página folha
------------------------	------------------

Definido na linha 260 do ficheiro [bm\\_noh.c](#).

#### 4.7.3.3 bm\_noh\_escrutina()

```
void bm_noh_escrutina (
    bm_noh * bmn,
    int nivel )
```

faz a impressão das chaves na página

##### Parâmetros

<a href="#">bm_noh</a>	bmn página raiz
<i>int</i>	nivel nivel da arvore

---

Definido na linha 185 do ficheiro [bm\\_noh.c](#).

#### 4.7.3.4 `bm_noh_inic()`

```
bm_noh * bm_noh_inic (
    int i,
    char eh_folha )
```

faz a inicialização de um nó

##### Parâmetros

<i>int</i>	i grau da página
<i>char</i>	eh_folha flag para saber se a página é interna ou externa

Definido na linha 19 do ficheiro [bm\\_noh.c](#).

#### 4.7.3.5 `bm_noh_insere()`

```
void bm_noh_insere (
    bm_noh * bmn,
    int chave )
```

faz a injeção de uma chave na página

##### Parâmetros

<i>bm_noh</i>	bmn página
<i>int</i>	chave chave a ser inserida

```
for (j = 0; j < bmn->nchaves; j++) printf("%d ", bmn->chaves[j]); printf("\n");
```

Definido na linha 139 do ficheiro [bm\\_noh.c](#).

#### 4.7.3.6 `bm_noh_ledisc()`

```
s_artigo * bm_noh_ledisc (
    bm_noh * bmn,
    int chave )
```

ler conteúdo do disco

---

## Parâmetros

<i>bm_noh</i>	bmn página
<i>int</i>	chave chave para a identificação no disco

Definido na linha 311 do ficheiro [bm\\_noh.c](#).

**4.7.3.7 bm\_noh\_pesquisa()**

```
bm_noh * bm_noh_pesquisa (
    bm_noh * bmn,
    int chave )
```

pesquisa uma chave na árvore

## Parâmetros

<i>bm_noh</i>	bmn página raiz
<i>int</i>	chave chave a ser buscada

Definido na linha 236 do ficheiro [bm\\_noh.c](#).

**4.7.3.8 bm\_noh\_pesquisa\_folha()**

```
bm_noh * bm_noh_pesquisa_folha (
    bm_noh * bmn,
    int chave )
```

pesquisa uma chave na árvore

## Parâmetros

<i>bm_noh</i>	bmn página raiz
<i>int</i>	chave chave a ser buscada

Definido na linha 217 do ficheiro [bm\\_noh.c](#).

**4.7.3.9 bm\_noh\_populaart()**

```
void bm_noh_populaart (
    int chave,
    s_artigo * art )
```

#### 4.7.3.10 bm\_noh\_split()

```
void bm_noh_split (
    bm_noh * bmn,
    bm_noh * y,
    int i )
```

realiza a partição de uma página folha

##### Parâmetros

<i>bm_noh</i>	bmn
<i>bm_noh</i>	y
<i>int</i>	i

Definido na linha 39 do ficheiro [bm\\_noh.c](#).

#### 4.7.3.11 bm\_noh\_split\_int()

```
bm_noh * bm_noh_split_int (
    bm_noh * bmn,
    bm_noh * y,
    bm_noh * z )
```

realiza a partição de uma página interna

##### Parâmetros

<i>bm_noh</i>	bmn
<i>bm_noh</i>	y
<i>int</i>	i

Definido na linha 78 do ficheiro [bm\\_noh.c](#).

#### 4.7.3.12 padding()

```
void padding (
    char ch,
    int n )
```

faz a impressão de espaços para a função escrutina

Definido na linha 173 do ficheiro [bm\\_noh.c](#).

---

## 4.8 bm\_noh.h

[Ir para a documentação deste ficheiro.](#)

```

00001 /*
00002  * alunos: joilnen leite, daniel morais, matheus silva
00003  *
00004  *
00005  */
00006 #ifndef BM_NOH_
00007 #define BM_NOH_
00008
00010 #define NOME_ARQ "bmais_arquivo"
00012 #define SUFIXO_BM ".bm"
00013
00014 typedef struct artigo
00015 {
00016     int id;
00017     int ano;
00018     char autor[200];
00019     char titulo[200];
00020     char revista[200];
00021     char DOI[20];
00022     char palavraChave[200];
00023     char invalido;
00024 } s_artigo;
00025
00026 typedef struct bm_noh
00027 {
00028     int *chaves, nchaves;
00029     int mgra;
00030     struct bm_noh **filhos;
00031     struct bm_noh *pai;
00032     long **dfilhos;
00033     char eh_folha;
00034 } bm_noh;
00035
00036 bm_noh *bm_noh_inic(int i, char eh_folha);
00037
00038 void bm_noh_split(bm_noh *bmn, bm_noh *y, int i);
00039 bm_noh *bm_noh_split_int(bm_noh *bmn, bm_noh *y, bm_noh *z);
00040
00041 void bm_noh_insere(bm_noh *bmn, int chave);
00042
00043 void padding(char ch, int n);
00044 void bm_noh_escrutina(bm_noh *bmn, int nivel);
00045
00046 bm_noh *bm_noh_pesquisa(bm_noh *bmn, int k);
00047 bm_noh *bm_noh_pesquisa_folha(bm_noh *bmn, int chave);
00048 int bm_noh_contem(bm_noh *bmn, int chave);
00049
00050 void bm_noh_escrevedisc(bm_noh *bmn, s_artigo *art);
00051 s_artigo *bm_noh_ledisc(bm_noh *bmn, int chave);
00052
00053 void bm_noh_populaart(int chave, s_artigo *art);
00054
00055
00056
00057 #define MAXFILHOS(m) 2 * m
00058 #define MAXCHAVES(m) MAXFILHOS(m) - 1
00059
00060 #endif

```

## 4.9 Referência ao ficheiro src/checklist.c

```

#include <stdio.h>
#include <string.h>
#include "j curses.h"
#include "checklist.h"

```

### Funções

- void [tela\\_checklist](#) (int i)



## 4.9.1 Documentação das funções

### 4.9.1.1 tela\_checklist()

```
void tela_checklist (
    int i )
```

mostra itens da avaliacao

#### Parâmetros

<i>i</i>	especifica que itens estao selecionados
----------	---

Definido na linha 13 do ficheiro [checklist.c](#).

## 4.10 checklist.c

[Ir para a documentação deste ficheiro.](#)

```
00001 /*
00002  * aluno: joilnen leite, daniel morais, matheus silva
00003  */
00004 #include <stdio.h>
00005 #include <string.h>
00006
00007 #include "j curses.h"
00008 #include "checklist.h"
00009
00013 void tela_checklist(int i)
00014 {
00015     int l = 8, c = 24, bo = 1UL;
00016     static unsigned int cklst_estado = 0;
00017     cklst_estado |= i ;
00018
00019     CKLSITEM(l++, c + 11, 0, S_UNDERL"avaliacao"S_NORM);
00020     CKLSITEM(l++, c, 0, " ");
00021     CKLSITEM(l++, c, cklst_estado & bo, ITEM_1); bo <= 1;
00022     CKLSITEM(l++, c, cklst_estado & bo, ITEM_2); bo <= 1;
00023     CKLSITEM(l++, c, cklst_estado & bo, ITEM_3); bo <= 1;
00024     CKLSITEM(l++, c, cklst_estado & bo, ITEM_4); bo <= 1;
00025     CKLSITEM(l++, c, cklst_estado & bo, ITEM_5); bo <= 1;
00026     CKLSITEM(l++, c, cklst_estado & bo, ITEM_6); bo <= 1;
00027
00028     CMR1;
00029 }
00030
00031
```

## 4.11 Referência ao ficheiro src/checklist.h

### Macros

- #define [ITEM\\_1](#) "1 impressao dos artigos"
- #define [ITEM\\_2](#) "2 busca por id"
- #define [ITEM\\_3](#) "3 insercao de novo [artigo](#)"
- #define [ITEM\\_4](#) "4 remocao de um [artigo](#)"
- #define [ITEM\\_5](#) "5 listagem de todos os art. em ordem (de um id)"
- #define [ITEM\\_6](#) "6 testes"

## Enumerações

- enum {  
    I\_NONE, I\_1, I\_2, I\_3,  
    I\_4, I\_5, I\_6, I\_MAX }

## Funções

- void [tela\\_checklist](#) (int i)

### 4.11.1 Documentação das macros

#### 4.11.1.1 ITEM\_1

```
#define ITEM_1 "1 impressao dos artigos"
```

itens da avalicao mostrados na tela

Definido na linha [17](#) do ficheiro [checklist.h](#).

#### 4.11.1.2 ITEM\_2

```
#define ITEM_2 "2 busca por id"
```

Definido na linha [18](#) do ficheiro [checklist.h](#).

#### 4.11.1.3 ITEM\_3

```
#define ITEM_3 "3 insercao de novo artigo"
```

Definido na linha [19](#) do ficheiro [checklist.h](#).

#### 4.11.1.4 ITEM\_4

```
#define ITEM_4 "4 remocao de um artigo"
```

Definido na linha [20](#) do ficheiro [checklist.h](#).

---

#### 4.11.1.5 ITEM\_5

```
#define ITEM_5 "5 listagem de todos os art. em ordem (de um id) "
```

Definido na linha 21 do ficheiro [checklist.h](#).

#### 4.11.1.6 ITEM\_6

```
#define ITEM_6 "6 testes"
```

Definido na linha 22 do ficheiro [checklist.h](#).

### 4.11.2 Documentação dos valores da enumeração

#### 4.11.2.1 anonymous enum

```
anonymous enum
```

usado na operacao de bits na mostra de itens da avalicao

Valores de enumerações

I_NONE	
I_1	
I_2	
I_3	
I_4	
I_5	
I_6	
I_MAX	

Definido na linha 25 do ficheiro [checklist.h](#).

### 4.11.3 Documentação das funções

#### 4.11.3.1 tela\_checklist()

```
void tela_checklist (  
    int i )
```

mostra itens da avaliacao

---

### Parâmetros

<i>i</i>	especifica que itens estão selecionados
----------	---

Definido na linha 13 do ficheiro `checklist.c`.

## 4.12 checklist.h

[Ir para a documentação deste ficheiro.](#)

```
00001 /*
00002  * aluno: joilnen leite, daniel morais, matheus silva
00003  */
00004 #ifndef CHECKLIST_H_
00005 #define CHECKLIST_H_
00006
00007 /*
00008  * aluno: joilnen leite
00009  */
00010
00014 void tela_checklist(int i);
00015
00017 #define ITEM_1 "1 impressao dos artigos"
00018 #define ITEM_2 "2 busca por id"
00019 #define ITEM_3 "3 insercao de novo artigo"
00020 #define ITEM_4 "4 remocao de um artigo"
00021 #define ITEM_5 "5 listagem de todos os art. em ordem (de um id)"
00022 #define ITEM_6 "6 testes"
00023
00025 enum {I_NONE, I_1, I_2, I_3, I_4, I_5, I_6, I_MAX};
00026
00027 #endif
00028
00029
```

## 4.13 Referência ao ficheiro `src/erro.h`

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

### Funções

- void `erro` (const char \*f, const char \*m)

### 4.13.1 Documentação das funções

#### 4.13.1.1 `erro()`

```
void erro (
    const char * f,
    const char * m )
```

Definido na linha 5 do ficheiro `erro.h`.

---

## 4.14 erro.h

[Ir para a documentação deste ficheiro.](#)

```
00001 #ifndef ERRO_H
00002 #define ERRO_H
00003 #include <stdlib.h>
00004
00005 void erro(const char *f, const char *m)
00006 {
00007
00008     /****
00009     fprintf(stderr, "\n=== * erro\nfuncao: ");
00010     fprintf(stderr, f);
00011     fprintf(stderr, "\nmensagem: ");
00012     fprintf(stderr, m);
00013     fprintf(stderr, "\n=== * ===\n\n");
00014     ****/
00015     exit(-1);
00016 }
00017
00018 #endif
```

## 4.15 Referência ao ficheiro src/jcurses.h

### Macros

- #define S\_LTELA "\033[H\033[J"
  - #define S\_CM "\033[24;62H"
  - #define S\_CMR "\033[24;58H"
  - #define S\_UNDERL "\033[1m"
  - #define S\_PISCA "\033[5m"
  - #define S\_PISCARAP "\033[6m"
  - #define S\_MAGENTAB "\033[45m"
  - #define S\_PRETO "\033[30m"
  - #define S\_PRETOB "\033[40m"
  - #define S\_BRANCO "\033[37m"
  - #define S\_BRANCOB "\033[47m"
  - #define S\_CINZA "\033[36m"
  - #define S\_CINZAB "\033[46m"
  - #define S\_AZUL "\033[34m"
  - #define S\_AZULB "\033[44m"
  - #define S\_VERM "\033[31m"
  - #define S\_VERMB "\033[41m"
  - #define S\_VERD "\033[32m"
  - #define S\_VERDB "\033[42m"
  - #define S\_INV "\033[7m"
  - #define S\_NORM "\033[0m"
  - #define S\_INFO "\033[24;0H\033[7m"
  - #define S\_INFO\_ST "\033[1;65H"
  - #define S\_INFO\_ST\_1 "\033[2;65H"
  - #define S\_CARD "\033[0;60H"
  - #define S\_CHKLIST "\033[%d;%dH%s\033[0m"
  - #define S\_CHKLIST\_DONE "\033[46m\033[30m\033[%d;%dH%s\033[0m"
  - #define S\_INFO2 "\033[46m\033[30m\033[24;0H"
  - #define S\_LIST "\033[%d;78H%d"
  - #define S\_FILEN "\033[0;55H"
  - #define S\_VENN S\_CHKLIST
  - #define S\_TEST "\033[1m"
-

- `#define S_NADA ""`
- `#define LTELA printf(S_LTELA)`
- `#define TIPO_DE_TERMINAL TCOLOR`
- `#define INFO(x) printf("%s%s%s", S_INFO, x, S_NORM)`
- `#define INFO2(x) printf("%s%s%s", S_INFO2, x, S_NORM)`
- `#define INFO_ST(x) printf("%s%s%s%s", S_INV, S_INFO_ST, x, S_NORM)`
- `#define CM printf("%s", m); fflush(stdout); getc(stdin)`
- `#define CMR(x) printf("%s", S_CMR"q + * tecla enter *"); fflush(stdout); x = getc(stdin)`
- `#define CMR1 printf("%s", S_CMR"++ tecla enter *"); fflush(stdout); getc(stdin)`
- `#define CKLSITEM(l, c, b, x)`
- `#define INFO_FILE printf("%s%s%sf:%s%s", S_INFO_ST, __FILE__, S_INFO_ST_1, __FUNCTION__↵, S_NORM); fflush(stdout);`
- `#define INFO_CARD(x) printf("%scardinalidade: %d%s", S_CARD, x, S_NORM)`
- `#define S_FORMAT "\033[%d;%dH%s%s%\033[0m"`
- `#define PNAPOS(l, c, a1, a2, m) printf(S_FORMAT, l, c, a1, a2, m)`

### 4.15.1 Documentação das macros

#### 4.15.1.1 CKLSITEM

```
#define CKLSITEM(
    l,
    c,
    b,
    x )
```

**Valor:**

```
if (b) printf(S_CHKLSIT_DONE, l, c, x); \
else printf(S_CHKLSIT, l, c, x)
```

Definido na linha 59 do ficheiro [jcurses.h](#).

#### 4.15.1.2 CM

```
#define CM printf("%s", m); fflush(stdout); getc(stdin)
```

Definido na linha 56 do ficheiro [jcurses.h](#).

#### 4.15.1.3 CMR

```
#define CMR(
    x ) printf("%s", S_CMR"q + * tecla enter *"); fflush(stdout); x = getc(stdin)
```

Definido na linha 57 do ficheiro [jcurses.h](#).

#### 4.15.1.4 CMR1

```
#define CMR1 printf("%s", S_CMR"++ tecla enter *"); fflush(stdout); getc(stdin)
```

Definido na linha 58 do ficheiro [jcurses.h](#).

#### 4.15.1.5 INFO

```
#define INFO(  
    x ) printf("%s%s%s", S_INFO, x, S_NORM)
```

Definido na linha 53 do ficheiro [jcurses.h](#).

#### 4.15.1.6 INFO2

```
#define INFO2(  
    x ) printf("%s%s%s", S_INFO2, x, S_NORM)
```

Definido na linha 54 do ficheiro [jcurses.h](#).

#### 4.15.1.7 INFO\_CARD

```
#define INFO_CARD(  
    x ) printf("%scardinalidade:  %d%s", S_CARD, x, S_NORM)
```

Definido na linha 63 do ficheiro [jcurses.h](#).

#### 4.15.1.8 INFO\_FILE

```
#define INFO_FILE printf("%s%s%sf:%s%s", S_INFO_ST, __FILE__, S_INFO_ST_1, __FUNCTION__↵  
, S_NORM); fflush(stdout);
```

Definido na linha 62 do ficheiro [jcurses.h](#).

#### 4.15.1.9 INFO\_ST

```
#define INFO_ST(  
    x ) printf("%s%s%s%s", S_INV, S_INFO_ST, x, S_NORM)
```

Definido na linha 55 do ficheiro [jcurses.h](#).

---

#### 4.15.1.10 LTELA

```
#define LTELA printf(S_LTELA)
```

Definido na linha [51](#) do ficheiro [jcurses.h](#).

#### 4.15.1.11 PNAPOS

```
#define PNAPOS(  
    l,  
    c,  
    a1,  
    a2,  
    m ) printf(S_FORMAT, l, c, a1, a2, m)
```

Definido na linha [65](#) do ficheiro [jcurses.h](#).

#### 4.15.1.12 S\_AZUL

```
#define S_AZUL "\033[34m"
```

Definido na linha [30](#) do ficheiro [jcurses.h](#).

#### 4.15.1.13 S\_AZULB

```
#define S_AZULB "\033[44m"
```

Definido na linha [31](#) do ficheiro [jcurses.h](#).

#### 4.15.1.14 S\_BRANCO

```
#define S_BRANCO "\033[37m"
```

Definido na linha [26](#) do ficheiro [jcurses.h](#).

#### 4.15.1.15 S\_BRANCOB

```
#define S_BRANCOB "\033[47m"
```

Definido na linha [27](#) do ficheiro [jcurses.h](#).

---



#### 4.15.1.16 S\_CARD

```
#define S_CARD "\033[0;60H"
```

Definido na linha 41 do ficheiro `jcurses.h`.

#### 4.15.1.17 S\_CHKLST

```
#define S_CHKLST "\033[%d;%dH%s\033[0m"
```

Definido na linha 42 do ficheiro `jcurses.h`.

#### 4.15.1.18 S\_CHKLST\_DONE

```
#define S_CHKLST_DONE "\033[46m\033[30m\033[%d;%dH%s\033[0m"
```

Definido na linha 43 do ficheiro `jcurses.h`.

#### 4.15.1.19 S\_CINZA

```
#define S_CINZA "\033[36m"
```

Definido na linha 28 do ficheiro `jcurses.h`.

#### 4.15.1.20 S\_CINZAB

```
#define S_CINZAB "\033[46m"
```

Definido na linha 29 do ficheiro `jcurses.h`.

#### 4.15.1.21 S\_CM

```
#define S_CM "\033[24;62H"
```

Definido na linha 18 do ficheiro `jcurses.h`.

---

#### 4.15.1.22 S\_CMR

```
#define S_CMR "\033[24;58H"
```

Definido na linha 19 do ficheiro [jcurses.h](#).

#### 4.15.1.23 S\_FILEN

```
#define S_FILEN "\033[0;55H"
```

Definido na linha 46 do ficheiro [jcurses.h](#).

#### 4.15.1.24 S\_FORMAT

```
#define S_FORMAT "\033[%d;%dH%s%s\033[0m"
```

Definido na linha 64 do ficheiro [jcurses.h](#).

#### 4.15.1.25 S\_INFO

```
#define S_INFO "\033[24;0H\033[7m"
```

Definido na linha 38 do ficheiro [jcurses.h](#).

#### 4.15.1.26 S\_INFO2

```
#define S_INFO2 "\033[46m\033[30m\033[24;0H"
```

Definido na linha 44 do ficheiro [jcurses.h](#).

#### 4.15.1.27 S\_INFO\_ST

```
#define S_INFO_ST "\033[1;65H"
```

Definido na linha 39 do ficheiro [jcurses.h](#).

---

#### 4.15.1.28 S\_INFO\_ST\_1

```
#define S_INFO_ST_1 "\033[2;65H"
```

Definido na linha 40 do ficheiro `jcurses.h`.

#### 4.15.1.29 S\_INV

```
#define S_INV "\033[7m"
```

Definido na linha 36 do ficheiro `jcurses.h`.

#### 4.15.1.30 S\_LIST

```
#define S_LIST "\033[%d;78H%d"
```

Definido na linha 45 do ficheiro `jcurses.h`.

#### 4.15.1.31 S\_LTELA

```
#define S_LTELA "\033[H\033[J"
```

marcos que criei baseado no que pesquisei sobre formatacao e codificacao de terminal vt100 que eh a base dos emuladores de terminal do linux curses eh como chamam o tipo de software q faz isso curses e ncurses sao exemplos, aqui eh soh uma tentativa primaria que atende as necessidades apenas dessa avaliacao

Definido na linha 17 do ficheiro `jcurses.h`.

#### 4.15.1.32 S\_MAGENTAB

```
#define S_MAGENTAB "\033[45m"
```

Definido na linha 23 do ficheiro `jcurses.h`.

#### 4.15.1.33 S\_NADA

```
#define S_NADA ""
```

Definido na linha 49 do ficheiro `jcurses.h`.

---

#### 4.15.1.34 S\_NORM

```
#define S_NORM "\033[0m"
```

Definido na linha [37](#) do ficheiro [jcurses.h](#).

#### 4.15.1.35 S\_PISCA

```
#define S_PISCA "\033[5m"
```

Definido na linha [21](#) do ficheiro [jcurses.h](#).

#### 4.15.1.36 S\_PISCARAP

```
#define S_PISCARAP "\033[6m"
```

Definido na linha [22](#) do ficheiro [jcurses.h](#).

#### 4.15.1.37 S\_PRETO

```
#define S_PRETO "\033[30m"
```

Definido na linha [24](#) do ficheiro [jcurses.h](#).

#### 4.15.1.38 S\_PRETOB

```
#define S_PRETOB "\033[40m"
```

Definido na linha [25](#) do ficheiro [jcurses.h](#).

#### 4.15.1.39 S\_TEST

```
#define S_TEST "\033[1m"
```

Definido na linha [48](#) do ficheiro [jcurses.h](#).

---

#### 4.15.1.40 S\_UNDERL

```
#define S_UNDERL "\033[1m"
```

Definido na linha 20 do ficheiro `jcurses.h`.

#### 4.15.1.41 S\_VENN

```
#define S_VENN S_CHKLIST
```

Definido na linha 47 do ficheiro `jcurses.h`.

#### 4.15.1.42 S\_VERD

```
#define S_VERD "\033[32m"
```

Definido na linha 34 do ficheiro `jcurses.h`.

#### 4.15.1.43 S\_VERDB

```
#define S_VERDB "\033[42m"
```

Definido na linha 35 do ficheiro `jcurses.h`.

#### 4.15.1.44 S\_VERM

```
#define S_VERM "\033[31m"
```

Definido na linha 32 do ficheiro `jcurses.h`.

#### 4.15.1.45 S\_VERMB

```
#define S_VERMB "\033[41m"
```

Definido na linha 33 do ficheiro `jcurses.h`.

---

#### 4.15.1.46 TIPO\_DE\_TERMINAL

```
#define TIPO_DE_TERMINAL TCOLOR
```

Definido na linha 52 do ficheiro `jcurses.h`.

## 4.16 jcurses.h

[Ir para a documentação deste ficheiro.](#)

```
00001 #ifndef JOILNEN_CURSES_H
00002 #define JOILNEN_CURSES_H
00003
00004 /*
00005  * aluno: joilnen
00006 */
00007
00017 #define S_LTELA "\033[H\033[J"
00018 #define S_CM "\033[24;62H"
00019 #define S_CMR "\033[24;58H"
00020 #define S_UNDERL "\033[1m"
00021 #define S_PISCA "\033[5m"
00022 #define S_PISCARAP "\033[6m"
00023 #define S_MAGENTAB "\033[45m"
00024 #define S_PRETO "\033[30m"
00025 #define S_PRETOB "\033[40m"
00026 #define S_BRANCO "\033[37m"
00027 #define S_BRANCOB "\033[47m"
00028 #define S_CINZA "\033[36m"
00029 #define S_CINZAB "\033[46m"
00030 #define S_AZUL "\033[34m"
00031 #define S_AZULB "\033[44m"
00032 #define S_VERM "\033[31m"
00033 #define S_VERMB "\033[41m"
00034 #define S_VERD "\033[32m"
00035 #define S_VERDB "\033[42m"
00036 #define S_INV "\033[7m"
00037 #define S_NORM "\033[0m"
00038 #define S_INFO "\033[24;0H\033[7m"
00039 #define S_INFO_ST "\033[1;65H"
00040 #define S_INFO_ST_1 "\033[2;65H"
00041 #define S_CARD "\033[0;60H"
00042 #define S_CHKLIST "\033[%d;%dH\033[0m"
00043 #define S_CHKLIST_DONE "\033[46m\033[30m\033[%d;%dH\033[0m"
00044 #define S_INFO2 "\033[46m\033[30m\033[24;0H"
00045 #define S_LIST "\033[%d;78H%d"
00046 #define S_FILEN "\033[0;55H"
00047 #define S_VENN S_CHKLIST
00048 #define S_TEST "\033[1m"
00049 #define S_NADA ""
00050
00051 #define LTELA printf(S_LTELA)
00052 #define TIPO_DE_TERMINAL TCOLOR
00053 #define INFO(x) printf("%s%s", S_INFO, x, S_NORM)
00054 #define INFO2(x) printf("%s%s", S_INFO2, x, S_NORM)
00055 #define INFO_ST(x) printf("%s%s", S_INFO_ST, x, S_NORM)
00056 #define CM printf("%s", m); fflush(stdout); getc(stdin)
00057 #define CMR(x) printf("%s", S_CMR"q + * tecla enter *"); fflush(stdout); x = getc(stdin)
00058 #define CMR1 printf("%s", S_CMR"++ tecla enter *"); fflush(stdout); getc(stdin)
00059 #define CKLSITEM(l,c,b,x) \
00060     if (b) printf(S_CHKLIST_DONE, l, c, x); \
00061     else printf(S_CHKLIST, l, c, x)
00062 #define INFO_FILE printf("%s%sf:%s", S_INFO_ST, __FILE__, S_INFO_ST_1, __FUNCTION__, S_NORM);
00063     fflush(stdout);
00063 #define INFO_CARD(x) printf("%scardinalidade: %d", S_CARD, x, S_NORM)
00064 #define S_FORMAT "\033[%d;%dH\033[0m"
00065 #define PNAPOS(l,c,a1,a2, m) printf(S_FORMAT, l, c, a1, a2, m)
00066
00067 #endif
```

## 4.17 Referência ao ficheiro src/main.c

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

```
#include <string.h>
#include <time.h>
#include "bm.h"
#include "checklist.h"
#include "jcurses.h"
```

## Funções

- int [main](#) ()

### 4.17.1 Documentação das funções

#### 4.17.1.1 main()

```
int main ( )
```

programa que testa a b+

Definido na linha [46](#) do ficheiro [main.c](#).

## 4.18 main.c

[Ir para a documentação deste ficheiro.](#)

```
00001 /* Grupo 11
00002  * alunos: joilnen leite, daniel morais, matheus silva
00003  *
00004  *
00005  */
00006 #include <stdio.h>
00007 #include <stdlib.h>
00008 #include <string.h>
00009 #include <time.h>
00010
00011 #include "bm.h"
00012 #include "checklist.h"
00013 #include "jcurses.h"
00014
00015 /****
00016 static void atravessa(bm_noh *r) {
00017     int i;
00018     if (!r)
00019         return;
00020     for (i = 0; i < r->nchaves; ++i)
00021         printf("%d ", r->chaves[i]);
00022     printf("\n");
00023     for (i = 0; i < r->nchaves + 1; ++i)
00024         atravessa(r->filhos[i]);
00025 }
00026 ****/
00027
00028 static void popula_art(s_artigo *art)
00029 {
00030     int chave, ano;
00031     srand(time(NULL));
00032     chave = rand() % 99 + 1;
00033     ano = rand() % 2000 + 1;
00034
00035     art->id = chave;
00036     art->ano = ano;
00037     strcpy(art->autor, "batman");
```

---

```

00038     strcpy(art->titulo, "a vida batman");
00039     strcpy(art->revista, "revista batman");
00040     strcpy(art->DOI, "DOI-DOI-MUITO-0000");
00041     strcpy(art->palavraChave, "batman");
00042     art->invalido = 0;
00043 }
00044
00046 int main()
00047 {
00048     int chave_teste = 6;
00049     int num = 50;
00050     bm *bm = bm_inic(3);
00051
00052     LTELA;
00053     tela_checklist(0);
00054     tela_checklist(I_1);
00055     tela_checklist(I_2);
00056     tela_checklist(I_3);
00057
00058     s_artigo *a = (s_artigo *) malloc(sizeof(s_artigo));
00059
00060     popula_art(a);
00061     bm_inserere(bm, 1 , a);
00062     CMR1;
00063     LTELA;
00064     popula_art(a);
00065     bm_inserere(bm, 72, a);
00066     CMR1;
00067     LTELA;
00068     popula_art(a);
00069     bm_inserere(bm, 3 , a);
00070     CMR1;
00071     LTELA;
00072     popula_art(a);
00073     bm_inserere(bm, 4 , a);
00074     CMR1;
00075     LTELA;
00076     popula_art(a);
00077     bm_inserere(bm, 40, a);
00078     CMR1;
00079     LTELA;
00080     // popula_art(a);
00081     // bm_inserere(bm, 5 , a);
00082     // popula_art(a);
00083     // bm_inserere(bm, 6 , a);
00084     // popula_art(a);
00085     // bm_inserere(bm, 7 , a);
00086     // popula_art(a);
00087     // bm_inserere(bm, 84, a);
00088     // popula_art(a);
00089     // bm_inserere(bm, 8 , a);
00090     // popula_art(a);
00091     // bm_inserere(bm, 99, a);
00092     // popula_art(a);
00093     // bm_inserere(bm, 10, a);
00094     // popula_art(a);
00095     // bm_inserere(bm, 11, a);
00096     // popula_art(a);
00097     // bm_inserere(bm, 12, a);
00098     // popula_art(a);
00099     // bm_inserere(bm, 173, a);
00100     // popula_art(a);
00101     // bm_inserere(bm, 14, a);
00102     // popula_art(a);
00103     // bm_inserere(bm, 199, a);
00104     // popula_art(a);
00105     // bm_inserere(bm, 110, a);
00106     // popula_art(a);
00107     // bm_inserere(bm, 111, a);
00108     // popula_art(a);
00109     // bm_inserere(bm, 112, a);
00110     free(a);
00111
00112     printf("Atravessa a arvore construida:");
00113     bm_escrutina(bm);
00114     /*
00115     if (bm_pesquisa(bm, chave_teste))
00116         printf("\nPresente");
00117     else
00118         printf("\nN Presente");
00119     chave_teste = 15;
00120
00121     if (bm_pesquisa(bm, chave_teste))
00122         printf("\nPresente");
00123     else
00124         printf("\nN Presente");
00125     printf("\n");*/

```

---



```

00126
00127     /*int i;
00128     for (i = 0; i < bm->raiz->nchaves; i++)
00129         printf("raiz %d\n", bm->raiz->chaves[i]);*/
00130
00131     printf("\nA partir de qual numero devo percorrer? ");
00132     int x = 0;
00133     scanf("%d", &x);
00134     bm_lista(bm->raiz, x);
00135
00136     free(bm);
00137     printf("\nfim\n");
00138
00139     return 0;
00140 }

```

## 4.19 Referência ao ficheiro src/testa.h

```

#include "jcurses.h"
#include "testa_bm.h"
#include "checklist.h"

```

## 4.20 testa.h

[Ir para a documentação deste ficheiro.](#)

```

00001 /*
00002  * aluno: joilnen leite, daniel morais, matheus silva
00003  */
00004 #ifndef TESTA_H_
00005 #define TESTA_H_
00006
00007 #include "jcurses.h"
00010 #include "testa_bm.h"
00011 #include "checklist.h"
00012
00013 #endif
00014

```

## 4.21 Referência ao ficheiro src/testa\_bm.c

```

#include <stdio.h>
#include <string.h>
#include "jcurses.h"
#include "testa_bm.h"

```

### Funções

- void [tela\\_testa\\_bm](#) ()
- void [testa\\_arvore\\_bm](#) ()

### 4.21.1 Documentação das funções

#### 4.21.1.1 tela\_testa\_bm()

```
void tela_testa_bm ( )
```

inclusao da biblioteca implementada baseada em arvores red black

Definido na linha 14 do ficheiro [testa\\_bm.c](#).

#### 4.21.1.2 testa\_arvore\_bm()

```
void testa_arvore_bm ( )
```

funcao main de teste separada do codigo da biblioteca como especificado PNAPOS(20, 39, S\_VERMB, S\_←BRANCO, S\_TEST"A");

PNAPOS(20, 39, S\_PRETO, S\_PISCA, S\_PRETO"A"S\_NORM);

Definido na linha 37 do ficheiro [testa\\_bm.c](#).

## 4.22 testa\_bm.c

[Ir para a documentação deste ficheiro.](#)

```
00001 /*
00002  * aluno: joilnen leite, daniel morais, matheus silva
00003  */
00004
00008 #include <stdio.h>
00009 #include <string.h>
00010
00011 #include "jcurls.h"
00012 #include "testa_bm.h"
00013
00014 void tela_testa_bm()
00015 {
00016     int li = 8, co = 24, f;
00017     char mens[7][49] =
00018     {
00019         S_UNDERL"serah efetuado os testes:"S_NORM,
00020         "",
00021         "insercao e remocao na arvore",
00022         "nao foi pedido mas me ajudou a",
00023         "checar a correcao da arvore",
00024         "cada simbolo da base representa",
00025         "paginas folhas ou multiplos delas"
00026     };
00027     LTELA;
00028     for (f = 0; f < 7; f++)
00029         CKLSITEM(li++, co, 0, mens[f]);
00030     INFO_FILE(__FILE__);
00031     CMR1;
00032 }
00033
00037 void testa_arvore_bm()
00038 {
00039     int i = 20;
00040     char temp[20];
00041     while (i)
00042     {
00043         LTELA;
00044         sprintf(temp, "ad. %dpg", 21 - i);
00045         INFO(temp);
00046         CMR1;
00047         i--;
00048     }
00049 }
00050
00051 }
00052
00053
00054
```

## 4.23 Referência ao ficheiro src/testa\_bm.h

```
#include "bm.h"
#include "jurses.h"
```

### Funções

- void [tela\\_testa\\_bm](#) ()
- void [testa\\_arvore\\_bm](#) ()

### 4.23.1 Documentação das funções

#### 4.23.1.1 tela\_testa\_bm()

```
void tela_testa_bm ( )
```

declaracao da funcao de teste da arvore b+

inclusao da biblioteca implementada baseada em arvores red black

Definido na linha [14](#) do ficheiro [testa\\_bm.c](#).

#### 4.23.1.2 testa\_arvore\_bm()

```
void testa_arvore_bm ( )
```

funcao main de teste separada do codigo da biblioteca como especificado PNAPOS(20, 39, S\_VERMB, S\_↔BRANCO, S\_TEST"A");

PNAPOS(20, 39, S\_PRETO, S\_PISCA, S\_PRETO"A"S\_NORM);

Definido na linha [37](#) do ficheiro [testa\\_bm.c](#).

## 4.24 testa\_bm.h

[Ir para a documentação deste ficheiro.](#)

```
00001 /*
00002  * aluno: joilnen leite, daniel morais, matheus silva
00003  */
00004
00005 #ifndef TESTA_ARVORE_BM_
00006 #define TESTA_ARVORE_BM_
00007
00008 #include "bm.h"
00009 #include "jurses.h"
00010
00015 void tela_testa_bm();
00016 void testa_arvore_bm();
00017
00018 #endif
00019
00020
```

---



# Índice

- ano
    - artigo, [5](#)
  - artigo, [5](#)
    - ano, [5](#)
    - autor, [5](#)
    - DOI, [6](#)
    - id, [6](#)
    - invalido, [6](#)
    - palavraChave, [6](#)
    - revista, [6](#)
    - titulo, [6](#)
  - autor
    - artigo, [5](#)
  - bm, [7](#)
    - bm.h, [15](#)
    - felem, [7](#)
    - mgrau, [7](#)
    - raiz, [7](#)
  - bm.c
    - bm\_escrutina, [11](#)
    - bm\_inic, [12](#)
    - bm\_insere, [12](#)
    - bm\_insere\_nahfolha, [12](#)
    - bm\_lista, [13](#)
    - bm\_pesquisa, [13](#)
  - bm.h
    - bm, [15](#)
    - bm\_escrutina, [16](#)
    - bm\_inic, [16](#)
    - bm\_insere, [16](#)
    - bm\_insere\_nahfolha, [17](#)
    - bm\_lista, [17](#)
    - bm\_pesquisa, [17](#)
  - bm\_escrutina
    - bm.c, [11](#)
    - bm.h, [16](#)
  - bm\_inic
    - bm.c, [12](#)
    - bm.h, [16](#)
  - bm\_insere
    - bm.c, [12](#)
    - bm.h, [16](#)
  - bm\_insere\_nahfolha
    - bm.c, [12](#)
    - bm.h, [17](#)
  - bm\_lista
    - bm.c, [13](#)
    - bm.h, [17](#)
  - bm\_noh, [8](#)
    - bm\_noh.h, [27](#)
    - chaves, [8](#)
    - dfilhos, [8](#)
    - eh\_folha, [8](#)
    - filhos, [9](#)
    - mgrau, [9](#)
    - nchaves, [9](#)
    - pai, [9](#)
  - bm\_noh.c
    - bm\_noh\_contem, [19](#)
    - bm\_noh\_escrevedisc, [19](#)
    - bm\_noh\_escrutina, [19](#)
    - bm\_noh\_inic, [20](#)
    - bm\_noh\_insere, [20](#)
    - bm\_noh\_ledisc, [20](#)
    - bm\_noh\_pesquisa, [21](#)
    - bm\_noh\_pesquisa\_folha, [21](#)
    - bm\_noh\_split, [21](#)
    - bm\_noh\_split\_int, [22](#)
    - padding, [22](#)
  - bm\_noh.h
    - bm\_noh, [27](#)
    - bm\_noh\_contem, [28](#)
    - bm\_noh\_escrevedisc, [28](#)
    - bm\_noh\_escrutina, [28](#)
    - bm\_noh\_inic, [29](#)
    - bm\_noh\_insere, [29](#)
    - bm\_noh\_ledisc, [29](#)
    - bm\_noh\_pesquisa, [30](#)
    - bm\_noh\_pesquisa\_folha, [30](#)
    - bm\_noh\_populaart, [30](#)
    - bm\_noh\_split, [30](#)
    - bm\_noh\_split\_int, [31](#)
    - MAXCHAVES, [27](#)
    - MAXFILHOS, [27](#)
    - NOME\_ARQ, [27](#)
    - padding, [31](#)
    - s\_artigo, [27](#)
    - SUFIXO\_BM, [27](#)
  - bm\_noh\_contem
    - bm\_noh.c, [19](#)
    - bm\_noh.h, [28](#)
  - bm\_noh\_escrevedisc
    - bm\_noh.c, [19](#)
    - bm\_noh.h, [28](#)
  - bm\_noh\_escrutina
    - bm\_noh.c, [19](#)
    - bm\_noh.h, [28](#)
-

- bm\_noh\_inic
  - bm\_noh.c, [20](#)
  - bm\_noh.h, [29](#)
- bm\_noh\_insere
  - bm\_noh.c, [20](#)
  - bm\_noh.h, [29](#)
- bm\_noh\_ledisc
  - bm\_noh.c, [20](#)
  - bm\_noh.h, [29](#)
- bm\_noh\_pesquisa
  - bm\_noh.c, [21](#)
  - bm\_noh.h, [30](#)
- bm\_noh\_pesquisa\_folha
  - bm\_noh.c, [21](#)
  - bm\_noh.h, [30](#)
- bm\_noh\_populaart
  - bm\_noh.h, [30](#)
- bm\_noh\_split
  - bm\_noh.c, [21](#)
  - bm\_noh.h, [30](#)
- bm\_noh\_split\_int
  - bm\_noh.c, [22](#)
  - bm\_noh.h, [31](#)
- bm\_pesquisa
  - bm.c, [13](#)
  - bm.h, [17](#)
- chaves
  - bm\_noh, [8](#)
- checklist.c
  - tela\_checklist, [33](#)
- checklist.h
  - I\_1, [35](#)
  - I\_2, [35](#)
  - I\_3, [35](#)
  - I\_4, [35](#)
  - I\_5, [35](#)
  - I\_6, [35](#)
  - I\_MAX, [35](#)
  - I\_NONE, [35](#)
  - ITEM\_1, [34](#)
  - ITEM\_2, [34](#)
  - ITEM\_3, [34](#)
  - ITEM\_4, [34](#)
  - ITEM\_5, [34](#)
  - ITEM\_6, [35](#)
  - tela\_checklist, [35](#)
- CKLSITEM
  - jcurses.h, [38](#)
- CM
  - jcurses.h, [38](#)
- CMR
  - jcurses.h, [38](#)
- CMR1
  - jcurses.h, [38](#)
- dfilhos
  - bm\_noh, [8](#)
- DOI
  - artigo, [6](#)
- eh\_folha
  - bm\_noh, [8](#)
- erro
  - erro.h, [36](#)
- erro.h
  - erro, [36](#)
- felem
  - bm, [7](#)
- filhos
  - bm\_noh, [9](#)
- I\_1
  - checklist.h, [35](#)
- I\_2
  - checklist.h, [35](#)
- I\_3
  - checklist.h, [35](#)
- I\_4
  - checklist.h, [35](#)
- I\_5
  - checklist.h, [35](#)
- I\_6
  - checklist.h, [35](#)
- I\_MAX
  - checklist.h, [35](#)
- I\_NONE
  - checklist.h, [35](#)
- id
  - artigo, [6](#)
- INFO
  - jcurses.h, [39](#)
- INFO2
  - jcurses.h, [39](#)
- INFO\_CARD
  - jcurses.h, [39](#)
- INFO\_FILE
  - jcurses.h, [39](#)
- INFO\_ST
  - jcurses.h, [39](#)
- invalido
  - artigo, [6](#)
- ITEM\_1
  - checklist.h, [34](#)
- ITEM\_2
  - checklist.h, [34](#)
- ITEM\_3
  - checklist.h, [34](#)
- ITEM\_4
  - checklist.h, [34](#)
- ITEM\_5
  - checklist.h, [34](#)
- ITEM\_6
  - checklist.h, [35](#)
- jcurses.h
  - CKLSITEM, [38](#)

- CM, [38](#)
- CMR, [38](#)
- CMR1, [38](#)
- INFO, [39](#)
- INFO2, [39](#)
- INFO\_CARD, [39](#)
- INFO\_FILE, [39](#)
- INFO\_ST, [39](#)
- LTELA, [39](#)
- PNAPOS, [40](#)
- S\_AZUL, [40](#)
- S\_AZULB, [40](#)
- S\_BRANCO, [40](#)
- S\_BRANCOB, [40](#)
- S\_CARD, [40](#)
- S\_CHKLST, [41](#)
- S\_CHKLST\_DONE, [41](#)
- S\_CINZA, [41](#)
- S\_CINZAB, [41](#)
- S\_CM, [41](#)
- S\_CMR, [41](#)
- S\_FILEN, [42](#)
- S\_FORMAT, [42](#)
- S\_INFO, [42](#)
- S\_INFO2, [42](#)
- S\_INFO\_ST, [42](#)
- S\_INFO\_ST\_1, [42](#)
- S\_INV, [43](#)
- S\_LIST, [43](#)
- S\_LTELA, [43](#)
- S\_MAGENTAB, [43](#)
- S\_NADA, [43](#)
- S\_NORM, [43](#)
- S\_PISCA, [44](#)
- S\_PISCARAP, [44](#)
- S\_PRETO, [44](#)
- S\_PRETOB, [44](#)
- S\_TEST, [44](#)
- S\_UNDERL, [44](#)
- S\_VENN, [45](#)
- S\_VERD, [45](#)
- S\_VERDB, [45](#)
- S\_VERM, [45](#)
- S\_VERMB, [45](#)
- TIPO\_DE\_TERMINAL, [45](#)
- 
- LTELA
  - [jcurses.h](#), [39](#)
- 
- main
  - [main.c](#), [47](#)
- main.c
  - [main](#), [47](#)
- MAXCHAVES
  - [bm\\_noh.h](#), [27](#)
- MAXFILHOS
  - [bm\\_noh.h](#), [27](#)
- mgrau
  - [bm](#), [7](#)
- 
- [bm\\_noh](#), [9](#)
- 
- nchaves
  - [bm\\_noh](#), [9](#)
- NOME\_ARQ
  - [bm\\_noh.h](#), [27](#)
- 
- padding
  - [bm\\_noh.c](#), [22](#)
  - [bm\\_noh.h](#), [31](#)
- pai
  - [bm\\_noh](#), [9](#)
- palavraChave
  - [artigo](#), [6](#)
- PNAPOS
  - [jcurses.h](#), [40](#)
- 
- raiz
  - [bm](#), [7](#)
- revista
  - [artigo](#), [6](#)
- 
- s\_artigo
  - [bm\\_noh.h](#), [27](#)
- S\_AZUL
  - [jcurses.h](#), [40](#)
- S\_AZULB
  - [jcurses.h](#), [40](#)
- S\_BRANCO
  - [jcurses.h](#), [40](#)
- S\_BRANCOB
  - [jcurses.h](#), [40](#)
- S\_CARD
  - [jcurses.h](#), [40](#)
- S\_CHKLST
  - [jcurses.h](#), [41](#)
- S\_CHKLST\_DONE
  - [jcurses.h](#), [41](#)
- S\_CINZA
  - [jcurses.h](#), [41](#)
- S\_CINZAB
  - [jcurses.h](#), [41](#)
- S\_CM
  - [jcurses.h](#), [41](#)
- S\_CMR
  - [jcurses.h](#), [41](#)
- S\_FILEN
  - [jcurses.h](#), [42](#)
- S\_FORMAT
  - [jcurses.h](#), [42](#)
- S\_INFO
  - [jcurses.h](#), [42](#)
- S\_INFO2
  - [jcurses.h](#), [42](#)
- S\_INFO\_ST
  - [jcurses.h](#), [42](#)
- S\_INFO\_ST\_1
  - [jcurses.h](#), [42](#)
- S\_INV

jurses.h, 43  
S\_LIST  
    jurses.h, 43  
S\_LTELA  
    jurses.h, 43  
S\_MAGENTAB  
    jurses.h, 43  
S\_NADA  
    jurses.h, 43  
S\_NORM  
    jurses.h, 43  
S\_PISCA  
    jurses.h, 44  
S\_PISCARAP  
    jurses.h, 44  
S\_PRETO  
    jurses.h, 44  
S\_PRETOB  
    jurses.h, 44  
S\_TEST  
    jurses.h, 44  
S\_UNDERL  
    jurses.h, 44  
S\_VENN  
    jurses.h, 45  
S\_VERD  
    jurses.h, 45  
S\_VERDB  
    jurses.h, 45  
S\_VERM  
    jurses.h, 45  
S\_VERMB  
    jurses.h, 45  
src/bm.c, 11, 13  
src/bm.h, 15, 18  
src/bm\_noh.c, 18, 22  
src/bm\_noh.h, 26, 32  
src/checklist.c, 32, 33  
src/checklist.h, 33, 36  
src/erro.h, 36, 37  
src/jurses.h, 37, 46  
src/main.c, 46, 47  
src/testa.h, 49  
src/testa\_bm.c, 49, 50  
src/testa\_bm.h, 51  
SUFIXO\_BM  
    bm\_noh.h, 27  
  
tela\_checklist  
    checklist.c, 33  
    checklist.h, 35  
tela\_testa\_bm  
    testa\_bm.c, 49  
    testa\_bm.h, 51  
testa\_arvore\_bm  
    testa\_bm.c, 50  
    testa\_bm.h, 51  
testa\_bm.c  
    tela\_testa\_bm, 49  
  
    testa\_arvore\_bm, 50  
testa\_bm.h  
    tela\_testa\_bm, 51  
    testa\_arvore\_bm, 51  
TIPO\_DE\_TERMINAL  
    jurses.h, 45  
titulo  
    artigo, 6

---