Programa F.3 *Estrutura do dicionário para árvore B*

```
typedef long TipoChave;
typedef struct TipoRegistro {
  TipoChave Chave;
  /*outros componentes*/
} TipoRegistro;
typedef struct TipoPagina* TipoApontador;
typedef struct TipoPagina {
  short n;
  TipoRegistro r[MM];
  TipoApontador p[MM + 1];
} TipoPagina;
```

Programa F.4 *Função para inicializar uma árvore B*

```
void Inicializa(TipoApontador *Dicionario)
{  *Dicionario = NULL; }
```

Programa F.5 *Função para pesquisar na árvore B*

```
void Pesquisa(TipoRegistro *x, TipoApontador Ap)
{ long i = 1;
  if (Ap == NULL)
  { printf("TipoRegistro nao esta presente na arvore\n");
    return;
  }
  while (i < Ap->n && x->Chave > Ap->r[i-1].Chave) i++;
  if (x->Chave == Ap->r[i-1].Chave)
  { *x = Ap->r[i-1];
    return;
  }
  if (x->Chave < Ap->r[i-1].Chave)
  Pesquisa(x, Ap->p[i-1]);
  else Pesquisa(x, Ap->p[i]);
}
```

Programa F.6 *Primeiro refinamento do algoritmo Insere na árvore B*

```
void Ins(TipoRegistro Reg, TipoApontador Ap, short *Cresceu,
         TipoRegistro *RegRetorno, TipoApontador *ApRetorno)
{ long i = 1; long j;
  TipoApontador ApTemp;
  if (Ap == NULL)
  { *Cresceu = TRUE;
    Atribui Reg a RegRetorno;
    Atribui NULL a ApRetorno;
    return;
  }
  while (i < Ap-> n && Reg.Chave > Ap -> r[i-1].Chave)  i++;
```


Continuação do Programa F.6

```
  if (Reg.Chave == Ap -> r[i-1].Chave)
  { printf(" Erro: Registro ja esta presente\n");
    return;
  }
  if (Reg.Chave < Ap -> r[i-1].Chave)
  Ins(Reg, Ap -> p[i--], Cresceu, RegRetorno, ApRetorno);
  if (!*Cresceu) return;
  if (Numero de registros em Ap < mm)
  { Insere na pagina Ap e *Cresceu = FALSE;
    return;
  }
  /* Overflow: Pagina tem que ser dividida */
  Cria nova pagina ApTemp;
  Transfere metade dos registros de Ap para ApTemp;
  Atribui registro do meio a RegRetorno;
  Atribui ApTemp a ApRetorno;
}

void Insere(TipoRegistro Reg, TipoApontador *Ap)
{ Ins(Reg, *Ap, &Cresceu, &RegRetorno, &ApRetorno);
  if (Cresceu)
  { Cria nova pagina raiz  para RegRetorno e ApRetorno;
  }
}
```

Programa F.7 *Função InsereNaPagina*

```
void InsereNaPagina(TipoApontador Ap,
                    TipoRegistro Reg, TipoApontador ApDir)
{ short NaoAchouPosicao;
  int k;
  k = Ap->n;  NaoAchouPosicao = (k > 0);
  while (NaoAchouPosicao)
    { if (Reg.Chave >= Ap->r[k-1].Chave)
      { NaoAchouPosicao = FALSE;
        break;
      }
      Ap->r[k] = Ap->r[k-1];
      Ap->p[k+1] = Ap->p[k];
      k--;
      if (k < 1) NaoAchouPosicao = FALSE;
    }
  Ap->r[k] = Reg;
  Ap->p[k+1] = ApDir;
  Ap->n++;
}
```