

Programa D.31 Primeiro Refinamento da função *OrdeneExterno*

```

#define ORDEMINTERCAL 2
void OrdeneExterno()
{
    int NBlocos = 0;
    ArqEntradaTipo ArqEntrada, ArqSaida;
    ArqEntradaTipo[ORDEMINTERCAL] ArrArqEnt;
    short Fim;
    int Low, High, Lim;
    NBlocos = 0;
    ArqEntrada = abrir arquivo a ser ordenado;
    do /*Formacao inicial dos NBlocos ordenados */
    {
        NBlocos++;
        Fim = EnchePaginas(NBlocos, ArqEntrada);
        OrdeneInterno;
        ArqSaida = AbreArqSaida(NBlocos);
        DescarregaPaginas(ArqSaida);
        fclose(ArqSaida);
    } while (!Fim);
    fclose(ArqEntrada); Low = 0; High = NBlocos-1;
    while (Low < High) /* Intercalacao dos NBlocos ordenados */
    {
        Lim = Minimo(Low + ORDEMINTERCAL - 1, High);
        AbreArqEntrada(ArrArqEnt, Low, Lim);
        High++;
        ArqSaida = AbreArqSaida(High);
        Intercala(ArrArqEnt, Low, Lim, ArqSaida);
        fclose(ArqSaida);
        for (i = Low; i < Lim; i++)
        {
            fclose(ArrArqEnt[i]);
            Apague_Arquivo(ArrArqEnt[i]);
        }
        Low += ORDEMINTERCAL;
    }
    Mudar o nome do arquivo High para o nome fornecido pelo usuario;
}

```

Apêndice

Programas em C do
Capítulo 5**Programa E.1** Estrutura do tipo dicionário implementado como arranjo

```

#define MAXN 10
typedef long TipoChave;
typedef struct TipoRegistro {
    TipoChave Chave;
    /* outros componentes */
} TipoRegistro;
typedef int TipoIndice;
typedef struct TipoTabela {
    TipoRegistro Item[MAXN + 1];
    TipoIndice n;
} TipoTabela;

```

Programa E.2 Implementação das operações usando arranjo

```

void Inicializa(TipoTabela *T)
{
    T->n = 0;
}

TipoIndice Pesquisa(TipoChave x, TipoTabela *T)
{
    int i;
    T->Item[0].Chave = x; i = T->n + 1;
    do {i--;} while (T->Item[i].Chave != x);
    return i;
}

void Insere(TipoRegistro Reg, TipoTabela *T)
{
    if (T->n == MAXN)
        printf("Erro : tabela cheia\n");
    else { T->n++; T->Item[T->n] = Reg; }
}

```