Unoeste
Campus I Rua José Bongiovani, 700
Cidade Universitária CEP 19050 900
Fone (18) 3229 2003

Faculdade de Informática de Presidente Prudente
Fone (18) 3229 1060
www.unoeste.br/fipp
e-mail: fipp@fipp.unoeste.br

FIPP
FACULDADE DE INFORMÁTICA
DE PRESIDENTE PRUDENTE
CAMPUS I

Pesquisa e Ordenação

Prof. Francisco Assis da Silva

– Atividade Prática de Implementação: Métodos de Ordenação –

Trabalho em dupla

---

1) Implementação de todos os algoritmos de ordenação em Lista Encadeada.

Algoritmos estudados em sala de aula:

- ~~Inserção Direta e Inserção Binária;~~
- ~~Seleção Direta;~~
- ~~Bolha e Shake;~~
- ~~Shell;~~
- ~~Heap;~~
- ~~Quick (recursivo e não recursivo));~~
- ~~Fusão Direta (Merge) (duas implementações).~~

Algoritmos para serem pesquisados na literatura e implementados:

- ~~Counting;~~
- ~~Bucket;~~
- ~~Radix;~~
- ~~Comb;~~
- ~~Gnome;~~
- ~~Tim.~~

---

2) Implementação de todos os algoritmos de ordenação em Arquivo Binário.

Arquivos Binários (Ordenado, Ordem Reversa e Randômico)

OBS: Acrescentar nos métodos de ordenação em arquivos 2 variáveis: **comparações** e **movimentações**.

A seguinte tabela deverá ser gerada e gravada em um arquivo texto!!!

Métodos Ordenação	**Arquivo Ordenado**					**Arquivo em Ordem Reversa**					**Arquivo Randômico**				
	Comp. Prog. *	Comp. Equa. #	Mov. Prog. +	Mov. Equa. -	Tempo	Comp. Prog.	Comp. Equa.	Mov. Prog.	Mov. Equa.	Tempo	Comp. Prog.	Comp. Equa.	Mov. Prog.	Mov. Equa.	Tempo
~~Inserção Direta~~															
~~Inserção Binária~~															
~~Seleção~~															
~~Bolha~~															
~~Shake~~															
~~Shell~~															
~~Heap~~															
~~Quick (pivô~~															
~~Quick (pivô~~															
~~Merge 1ª implement~~															
~~Merge 2ª implement~~															
~~Counting~~															
~~Bucket~~															
~~Radix~~															
~~Comb~~															
~~Gnome~~															
~~Tim~~															

Obs: os arquivos devem conter pelo menos 1024 registros.

* Comp Prog. = é a quantidade de comparações que foram realizadas no algoritmo, por exemplo, “reg.getNumero() > xxxx”. Não conte coisas como “i < TL”, ou seja, somente quando envolver o campo “Numero” do arquivo.
# Comp Equa. = é o valor resultante das equações de complexidade. Observe que do algoritmo Shell em diante não possui.
+ Mov. Prog. = é a quantidade de movimentações no algoritmo, quando tiver uma permutação, conte 2 movimentações.
- Mov. Equa. = é o valor resultante das equações de complexidade. Observe que do algoritmo Shell em diante não possui.

Esboço para a implementação do trabalho (parte de arquivo):

```
class Registro
{
  public final int tf=1022;
  private int numero; //4 bytes
  private char lixo[] = new char[tf]; //2044 bytes

  public Registro(int numero)
  {
    this.numero=numero;
    for (int i=0 ; i<tf ; i++)
      lixo[i]='X';
  }

  public void gravaNoArq(RandomAccessFile arquivo)
  {
    try
    {
      arquivo.writeInt(numero);
      for(int i=0 ; i<tf ; i++)
        arquivo.writeChar(lixo[i]);
    }catch(IOException e){}
  }

  public void leDoArq(RandomAccessFile arquivo)
  {
    try
    {
      numero = arquivo.readInt();
      for(int i=0 ; i<tf ; i++)
        lixo[i]=arquivo.readChar();
    }catch(IOException e){}
  }

  static int length()
  {
    //int numero;                    4 bytes
    //char lixo[] = new char[tf]; 2044 bytes
    //-------------------------------------
    //                            2048 bytes
      return(2048);
  }
}
```

```
class Arquivo
{
  private String nomearquivo;
  private RandomAccessFile arquivo;
  private int comp, mov;

  public Arquivo(String nomearquivo) {...}
```

```
  public void copiaArquivo(RandomAccessFile arquivoOrigem){...}
  public RandomAccessFile getFile() {...}

  public void truncate(long pos) {...}
  public boolean eof() {...}
  public void seekArq(int pos) {...}
  public void filesize() {...}

  public void initComp() {...}
  public void initMov() {...}
  public int getComp() {...}
  public int getMov() {...}

  public void insercaoDireta() {...}
  //demais metodos de ordenacao

  public void geraArquivoOrdenado() {...}
  public void geraArquivoReverso() {...}
  public void geraArquivoRandomico() {...}
}
```

```
public class Principal
{
  Arquivo arqOrd, arqRev, arqRand, auxRev, auxRand;

  ...

  public void geraTabela()
  {
    arqOrd.geraArquivoOrdenado();
    arqRev.geraArquivoReverso();
    arqRand.geraArquivoRandomico();

    //... Insercao Direta ...

    //Arquivo Ordenado
    arqOrd.initComp();
    arqOrd.initMov();
    tini=System.currentTimeMillis(); //método para pegar a hora atual em milisegundos
    arqOrd.isercaoDireta();
    tfim=System.currentTimeMillis(); //método para pegar a hora atual em milisegundos
    compO=arqOrd.getComp();
    movO=arqOrd.getMov();
    ttotalO=tfim-tini;

    //Arquivo Reverso
    auxRev.copiaArquivo(arqRev.getFile()); //faz uma cópia do arquivo de arqRev
                                           //para auxRev para preservar o original
    auxRev.initComp();
    auxRev.initMov();
    tini=System.currentTimeMillis();
    auxRev.isercaoDireta();
    tfim=System.currentTimeMillis();
    ttotalRev=tfim-tini;
    compRev=auxRev.getComp();
    movRev= auxRev.getMov();
```

```
    //Arquivo Randomico
    auxRand.copiaArquivo(arqRand.getFile()); //faz uma cópia do arquivo de arqRand
                                             //para auxRand para preservar o original
    auxRand.initComp();
    auxRand.initMov();
    tini=System.currentTimeMillis();
    auxRand.isercaoDireta();
    tfim=System.currentTimeMillis();
    ttotalRand=tfim-tini;
    compRand=auxRand.getComp();
    movRand=auxRand.getMov();
    //grava na tabela informacoes os dados extraídos das execucoes do método
    //Insercao Direta
    gravaLinhaTabela(compO,
                     calculaCompInsDir(filesize()),
                     movO,
                     calculaMovInsDir(filesize()),
                     ttotalO, //tempo execução no arquivo Ordenado já convertido
                              //para segundos
                     ...
                     )
    //... fim Insercao Direta

    //e assim continua para os outros métodos de ordenacao!!!

  }
  ...

  public static void main(String args[])
  {
    Principal p = new Principal();
    p.geraTabela();
  }
}
```