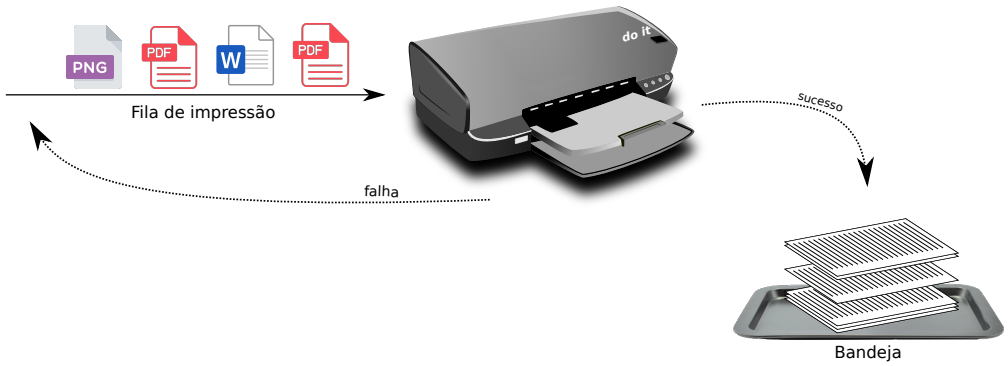
Algoritmos e Estruturas de Dados 1

2024/2 - Trabalho da Disciplina

Enunciado

O Departamento de Informática recebeu uma nova impressora, a novíssima *Náique Ultimate Printer Plus 2025 – Failproof Edition*! A figura abaixo apresenta um esquema de como a impressora funciona:



Quando um usuário manda imprimir um documento, ele não é impresso instantaneamente. Ao invés disso, o documento é apenas inserido na *fila de impressão* da impressora.

Quando o botão "*do it*" é pressionado (e apenas neste nomento!) a impressora retira o documento no início da fila de impressão e *tenta* imprimi-lo. Caso consiga imprimir o documento (sucesso), o documento impresso é automaticamente grampeado e é *empilhado* em sua *bandeja* – ficando em cima dos documentos que já estiverem nela. Caso não consiga imprimir o documento (falha), o documento é reinserido no *fim* da fila de impressão.

O usuário só poderá pegar um documento impresso ao removê-lo da bandeja, e ele sempre pegará o documento que estiver mais ao todo dela.

Como a memória da *Náique Ultimate Printer Plus 2025* é limitada, a fila de impressão tem capacidade para armazenar no máximo 20 documentos simultaneamente, enquanto a bandeja pode armazenar no máximo 10 documentos simultaneamante (independentemente do número de páginas).

Escreva um programa em C que simula a operação da impressora. Seu programa deve ler do usuário uma sequência de comandos, um a um, e os processar. Cada comando pode ser:

- `print [nome] [paginas] [prob]`: insere, na fila de impressão, um documento chamado `[nome]`, que tem `[paginas]` páginas, e que a impressora tem `[prob]` % de chance de imprimir. Como exemplo, o comando

  ```
  print declaracao de matricula.pdf 5 75
  ```

  insere na fila de impressão o documento `declaracao de matricula.pdf`, que tem 5 páginas, e cuja chance de ser impresso é de 75%.

  Note que o comando é dado em uma única linha, e que o nome do arquivo pode ou não conter espaços.

  Se a fila de impressão estiver cheia, imprima a mensagem `Erro: fila de impressao cheia.` e ignore o comando; caso contrário, imprima a mensagem `[nome] ([paginas] paginas) adicionado a fila.`.

- `doit`: botão "*do it*" é pressionado. Se a fila de impressão estiver vazia, imprima a mensagem `Erro: Fila de impressao vazia.`; caso contrário, se a bandeja estiver cheia, imprima a mensagem

`Erro: Bandeja cheia.` (e não altere a fila); caso contrário, a impressora tenta imprimir o documento no início da fila (a probabilidade de sucesso foi dada no comando `print` do respectivo documento).

Em caso de sucesso, insira o documento na bandeja e imprima a mensagem `[nome] ([paginas] paginas) impresso e adicionado a bandeja.`. Em caso de falha, insira o documento no final da fila e imprima a mensagem `[nome] ([paginas] paginas) falhou; reenfilerado.`.

Seu programa deve determinar se a impressão é um sucesso ou uma falha *aleatoriamente, de acordo com a probabilidade do documento ser impresso*. Por conter um fator aleatório, note que a saída do seu programa pode não ser a mesma dada nos exemplos!

- `pick`: retira e pega um documento da bandeja. Se a bandeja estiver vazia, imprima a mensagem `Erro: Bandeja vazia.`; caso contrário, retire o documento no topo da bandeja e imprima a mensagem `[nome] ([paginas] paginas) retirado da bandeja.`.
- `info`: mostra informações do sistema (sem alterar nada). Apenas imprima a mensagem `Ha [F] documentos na fila e [B] documentos na bandeja.` indicando que há `[F]` documentos na fila de impressão e `[B]` documentos na bandeja no momento.
- `F`: encerra a simulação. Ao encerrar, seu programa deve imprimir informações sobre os documentos que ainda ficaram na fila e na bandeja, conforme formato e ordem dados no exemplo abaixo.

Confira o exemplo de execução abaixo, onde o símbolo `>` indica a entrada do usuário (note que, devido ao fator aleatório, a saída do seu programa pode não ser exatamente a dada no exemplo, embora o *formato* da saída devará ser igual):

```
> info
Ha 0 documentos na fila e 0 documentos na bandeja.

> print declaracao de matricula.pdf 5 75
declaracao de matricula.pdf (5 paginas) adicionado a fila.

> print pBuZEGYXA6E.mp4 1 0
pBuZEGYXA6E.mp4 (1 paginas) adicionado a fila.

> info
Ha 2 documentos na fila e 0 documentos na bandeja.

> doit
declaracao de matricula.pdf (5 paginas) falhou; reenfilerado.

> doit
pBuZEGYXA6E.mp4 (1 paginas) falhou; reenfilerado.

> info
Ha 2 documentos na fila e 0 documentos na bandeja.

> doit
declaracao de matricula.pdf (5 paginas) impresso e adicionado a bandeja.

> print trabalho ED final-v2-agoravai.tex 10 50
trabalho ED final-v2-agoravai.tex (10 paginas) adicionado a fila.

> pick
declaracao de matricula.pdf (5 paginas) retirado da bandeja.
```

```
> pick
Erro: Bandeja vazia.

> doit
pBuZEGYXA6E.mp4 (1 paginas) falhou; reenfilerado.

> doit
trabalho ED final-v2-agoravai.tex (10 paginas) impresso e adicionado a bandeja.

> info
Ha 1 documentos na fila e 1 documentos na bandeja.

> F
=== fila de impressao ===
pBuZEGYXA6E.mp4 (1 paginas)
=== bandeja ===
trabalho ED final-v2-agoravai.tex (10 paginas)
```

Confira o segundo exemplo de execução abaixo:

```
> doit
Erro: Fila de impressao vazia.

> pick
Erro: Bandeja vazia.

> print documento1.txt 1 100
documento1.txt (1 paginas) adicionado a fila.

> print documento2.txt 1 100
documento2.txt (1 paginas) adicionado a fila.

> print documento3.txt 1 100
documento3.txt (1 paginas) adicionado a fila.

> print documento4.txt 1 100
documento4.txt (1 paginas) adicionado a fila.

> print documento5.txt 1 100
documento5.txt (1 paginas) adicionado a fila.

> print documento6.txt 1 100
documento6.txt (1 paginas) adicionado a fila.

> print documento7.txt 1 100
documento7.txt (1 paginas) adicionado a fila.

> print documento8.txt 1 100
documento8.txt (1 paginas) adicionado a fila.

> print documento9.txt 1 100
documento9.txt (1 paginas) adicionado a fila.
```

```
> print documento10.txt 1 100
documento10.txt (1 paginas) adicionado a fila.

> print documento11.txt 1 100
documento11.txt (1 paginas) adicionado a fila.

> print documento12.txt 1 100
documento12.txt (1 paginas) adicionado a fila.

> print documento13.txt 1 100
documento13.txt (1 paginas) adicionado a fila.

> print documento14.txt 1 100
documento14.txt (1 paginas) adicionado a fila.

> print documento15.txt 1 100
documento15.txt (1 paginas) adicionado a fila.

> print documento16.txt 1 100
documento16.txt (1 paginas) adicionado a fila.

> print documento17.txt 1 100
documento17.txt (1 paginas) adicionado a fila.

> print documento18.txt 1 100
documento18.txt (1 paginas) adicionado a fila.

> print documento19.txt 1 100
documento19.txt (1 paginas) adicionado a fila.

> print documento20.txt 1 100
documento20.txt (1 paginas) adicionado a fila.

> print documento21.txt 1 100
Erro: fila de impressao cheia.

> print documento22.txt 1 100
Erro: fila de impressao cheia.

> doit
documento1.txt (1 paginas) impresso e adicionado a bandeja.

> doit
documento2.txt (1 paginas) impresso e adicionado a bandeja.

> doit
documento3.txt (1 paginas) impresso e adicionado a bandeja.

> doit
documento4.txt (1 paginas) impresso e adicionado a bandeja.

> doit
documento5.txt (1 paginas) impresso e adicionado a bandeja.
```

```
> doit
documento6.txt (1 paginas) impresso e adicionado a bandeja.

> doit
documento7.txt (1 paginas) impresso e adicionado a bandeja.

> doit
documento8.txt (1 paginas) impresso e adicionado a bandeja.

> doit
documento9.txt (1 paginas) impresso e adicionado a bandeja.

> doit
documento10.txt (1 paginas) impresso e adicionado a bandeja.

> doit
Erro: Bandeja cheia.

> doit
Erro: Bandeja cheia.

> doit
Erro: Bandeja cheia.

> F
=== fila de impressao ===
documento11.txt (1 paginas)
documento12.txt (1 paginas)
documento13.txt (1 paginas)
documento14.txt (1 paginas)
documento15.txt (1 paginas)
documento16.txt (1 paginas)
documento17.txt (1 paginas)
documento18.txt (1 paginas)
documento19.txt (1 paginas)
documento20.txt (1 paginas)
=== bandeja ===
documento10.txt (1 paginas)
documento9.txt (1 paginas)
documento8.txt (1 paginas)
documento7.txt (1 paginas)
documento6.txt (1 paginas)
documento5.txt (1 paginas)
documento4.txt (1 paginas)
documento3.txt (1 paginas)
documento2.txt (1 paginas)
documento1.txt (1 paginas)
```

Implementação

O trabalho deve **obrigatoriamente** usar pilha(s) **e** fila(s) em sua solução. O trabalho deve conter os seguintes arquivos:

- `PE.h` e `PE.c`: definição e implementação de pilha usando como base um vetor ("pilha estática");
- `PD.h` e `PD.c`: definição e implementação de pilha usando como base uma lista ligada ("pilha dinâmica");
- `FE.h` e `FE.c`: definição e implementação de fila usando como base um vetor ("fila estática");
- `FD.h` e `FD.c`: definição e implementação de fila usando como base uma lista ligada ("fila dinâmica");
- `main.c`: programa principal. Deve incluir (via `#include`):
  - `PE.h` *ou* `PD.h`; *e*
  - `FE.h` *ou* `FD.h`.

O programa principal deve utilizar filas e pilhas como estruturas *abstratas* de dados. Em particular, deve ser possível "escolher" entre usar pilhas estáticas ou dinâmicas *apenas alterando os* `#include` *e recompilando de acordo*! Da mesma forma, deve ser possível "escolher" entre usar filas estáticas ou dinâmicas de maneira análoga (note que, desta forma, há um total de quatro "configurações" com as quais o trabalho deverá funcionar).

Independentemente da implementação, certifique-se que toda memória alocada por seu programa é desalocada ao final da sua execução.

Orientações

- O trabalho pode ser feito por equipes de *até* 2 (dois) estudantes;
- Submeta, via *Moodle*, um pacote (zip ou tar.gz) contendo os 9 arquivos citados acima, além de um arquivo de texto (txt) onde conste:
  - O nome de todos os integrantes da equipe;
  - Toda informação que a equipe julgar relevante para a correção (como *bugs* conhecidos, detalhes de implementação, escolhas de projeto, etc.)
- Comente adequadamente seus códigos para facilitar a correção.
- Atenção: a correção será parcialmente automatizada, e a saída do programa será testada com outras entradas além das fornecidas como exemplo. *Siga* ***fielmente*** *o formato de saída dado nos exemplos*, sob pena de grande redução da nota;
- Certifique-se que seu programa funciona antes de submetê-lo;
- O trabalho deve ser entregue até **15 de Dezembro de 2024, 23:59**, via *Moodle*. É suficiente que o trabalho seja submetido por apenas um estudante da equipe;
- Trabalhos detectados como cópia/plágio (de colegas, da internet ou de ferramentas de IA), ou comprados, receberão **todos** a nota 0 (**ZERO**) e estarão sujeitos a abertura de Processo Administrativo Disciplinar Discente.