

IMPLEMENTAÇÃO DE SISTEMA DE ARQUIVOS

GUSTAVO DE OLIVEIRA E KEVEN GUSTAVO

OBJETIVOS

- 
- 
- 01** MANIPULAR ARQUIVOS E DIRETORIOS ATRAVÉS DE OPERAÇÕES
 - 02** IMPLEMENTAR ALOCAÇÃO DE ARQUIVOS NO DISCO
 - 03** ALOCAÇÃO CONTÍGUA FIRST-FIT, BEST-FIT E WORST-FIT
 - 04** ALOCAÇÃO ENCADEADA

REFÉRENCIAS

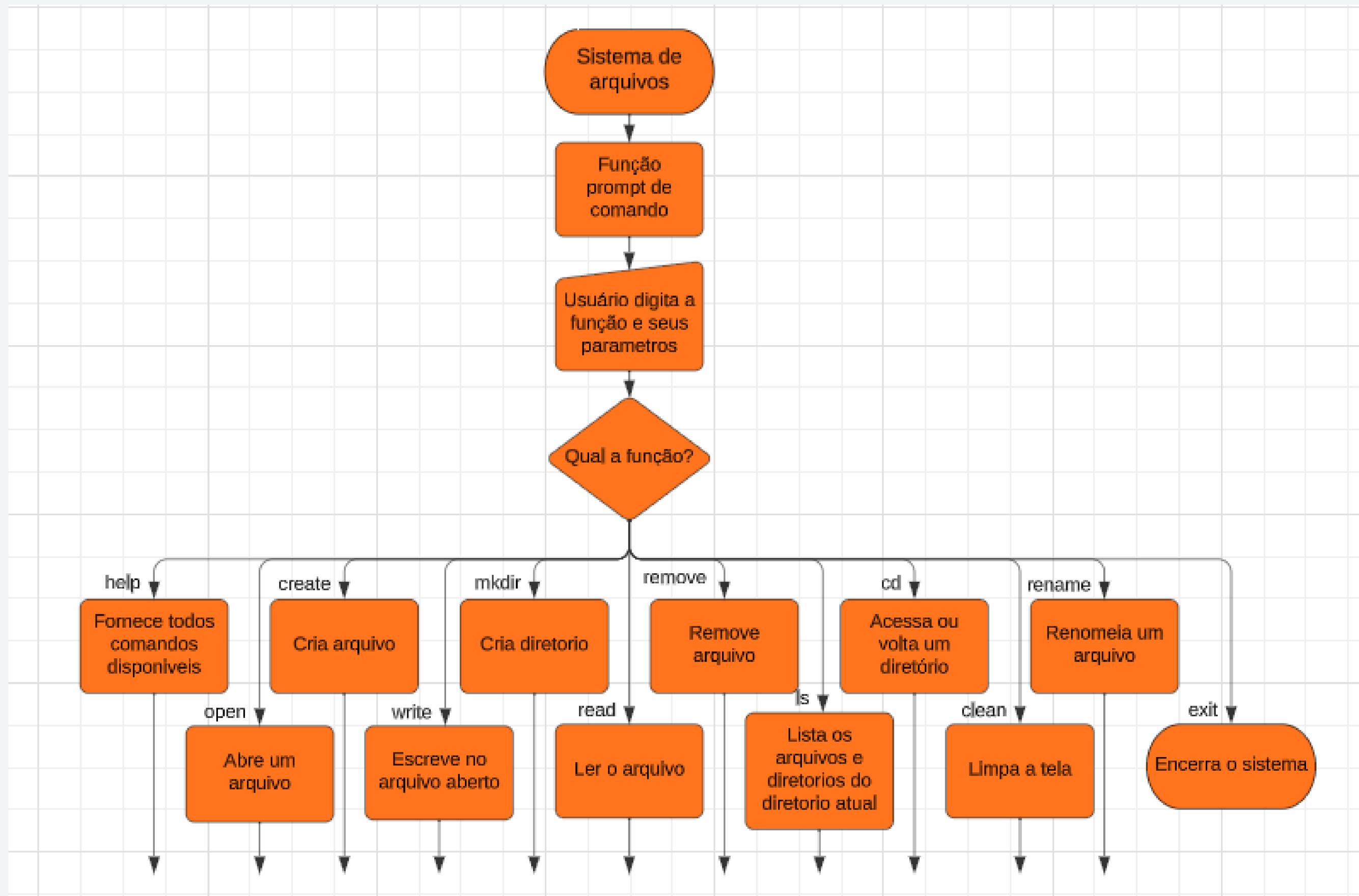
O código do gerenciador de arquivos utiliza como base de seu sistema o código fornecido pelo professor Thales Valente.



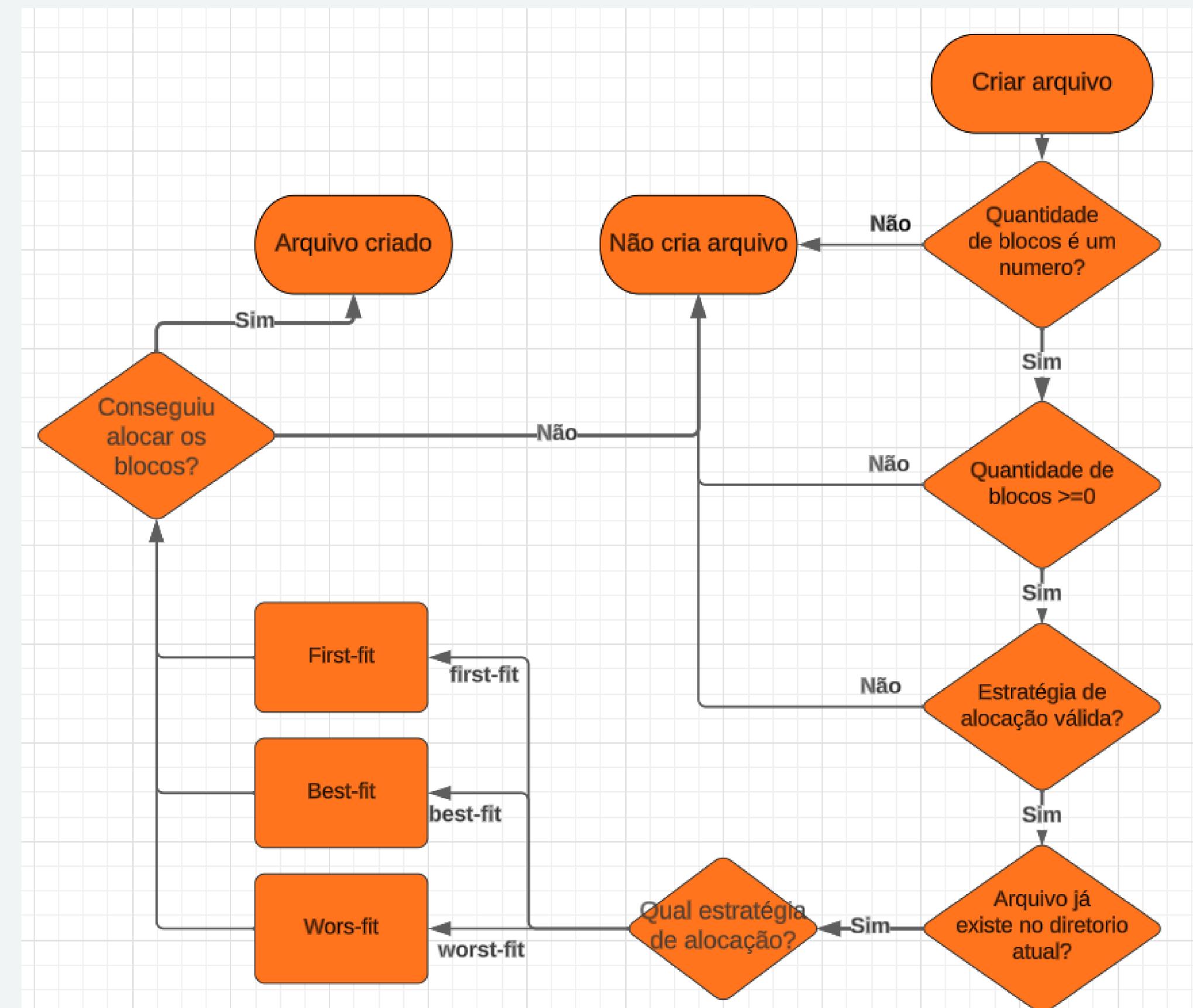
BIBLIOTECAS

No código é utilizado a biblioteca Anytree que é utilizada para a manipulação de estrutura de dados em árvore, no caso do nosso código foi utilizada para a realização da árvore de diretórios

FLUXOGRAMA - SISTEMA



CRIAR ARQUIVO - ALOCAÇÃO CONTÍGUA



CRIAR ARQUIVO - ALOCAÇÃO CONTIGUA

Para a criação de um arquivo no programa é preciso o usuário digitar “create” e separar por espaços os parâmetros:

- Nome do arquivo.
- Quantidades de blocos a serem alocados.
- Estratégia de alocação.

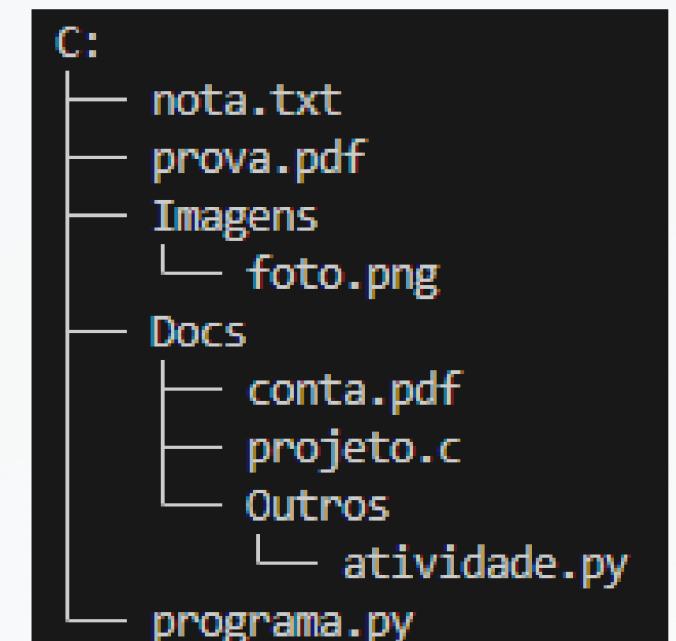
Exemplo de input do usuário:

```
create programa.py 3 first-fit
```

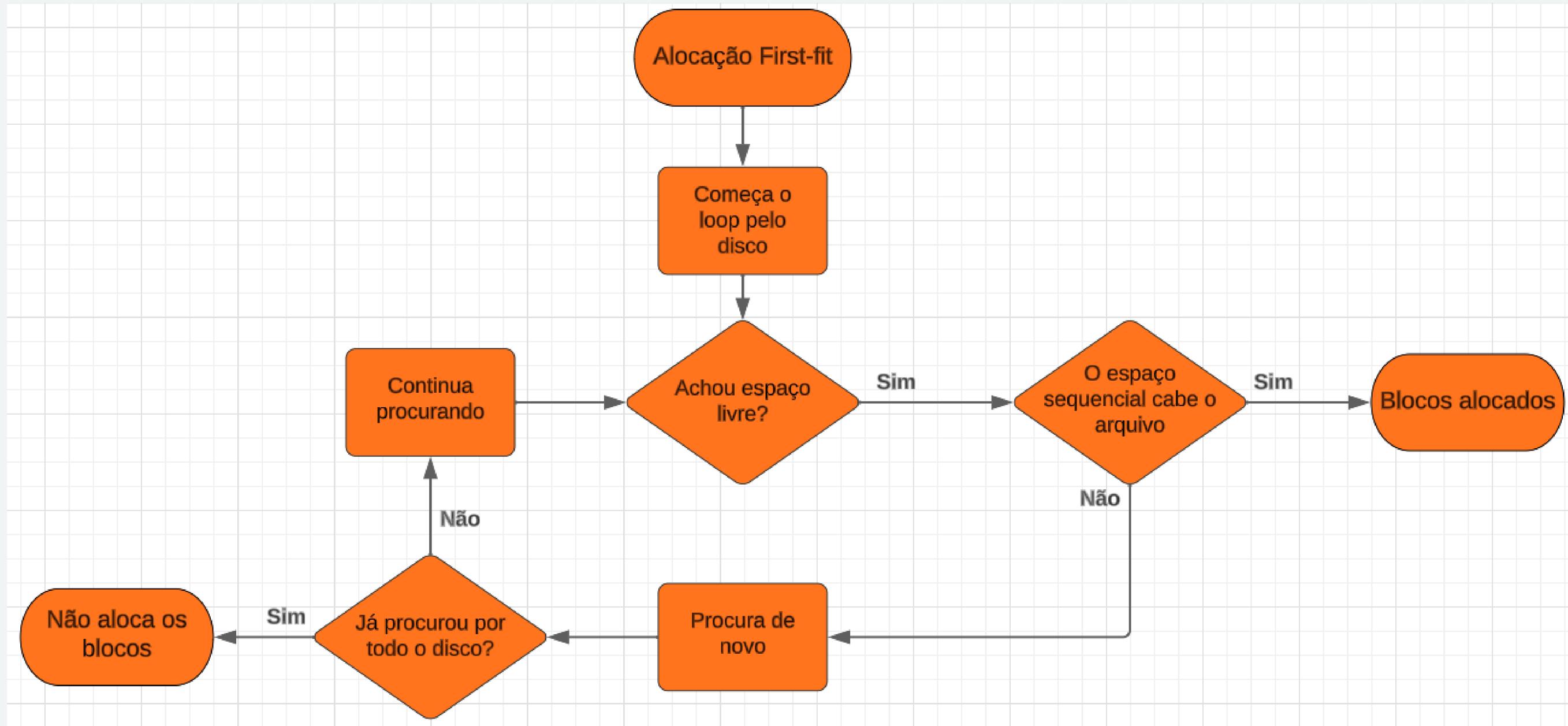
Exemplo de output do sistema:

```
Arquivo 'programa.py' criado com sucesso.
```

Como o arquivo é exibido no sistema de árvore:



ALOCAÇÃO CONTÍGUA - FIRST-FIT



ALOCAÇÃO CONTIGUA - FIRST-FIT

Situação inicial do disco:

```
Bloco 0: Alocado para 'nota.txt'  
Bloco 1: Livre  
Bloco 2: Livre  
Bloco 3: Livre  
Bloco 4: Alocado para 'foto.png'  
Bloco 5: Livre  
Bloco 6: Livre  
Bloco 7: Alocado para 'projeto.c'  
Bloco 8: Alocado para 'atividade.py'  
Bloco 9: Livre  
Bloco 10: Livre  
Bloco 11: Livre  
Bloco 12: Livre
```

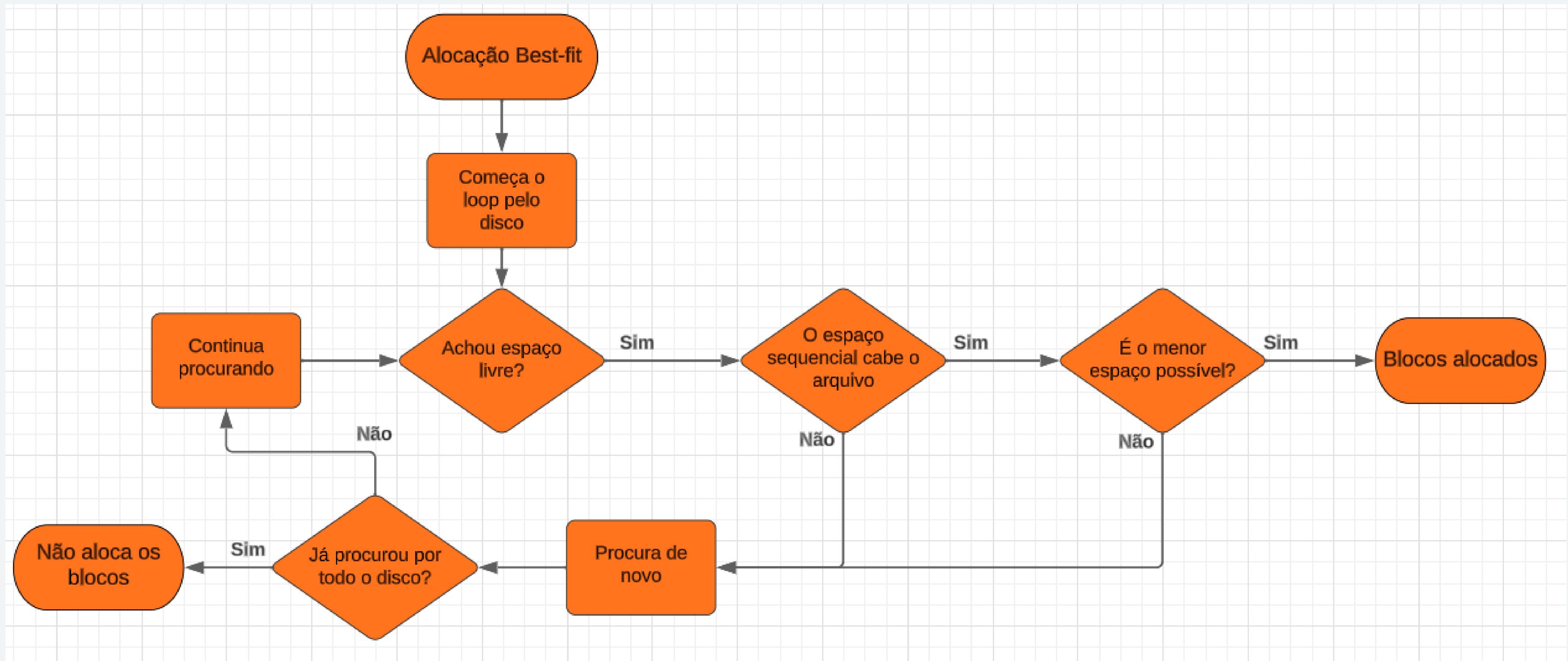
Input do usuário:

```
create programa.py 3 first-fit
```

Situação final do disco:

```
Bloco 0: Alocado para 'nota.txt'  
Bloco 1: Alocado para 'programa.py'  
Bloco 2: Alocado para 'programa.py'  
Bloco 3: Alocado para 'programa.py'  
Bloco 4: Alocado para 'foto.png'  
Bloco 5: Livre  
Bloco 6: Livre  
Bloco 7: Alocado para 'projeto.c'  
Bloco 8: Alocado para 'atividade.py'  
Bloco 9: Livre  
Bloco 10: Livre  
Bloco 11: Livre  
Bloco 12: Livre
```

ALOCAÇÃO CONTÍGUA - BEST-FIT



ALOCAÇÃO CONTIGUA - BEST-FIT

Situação inicial do disco:

```
Bloco 0: Alocado para 'nota.txt'  
Bloco 1: Livre  
Bloco 2: Livre  
Bloco 3: Livre  
Bloco 4: Alocado para 'foto.png'  
Bloco 5: Livre  
Bloco 6: Livre  
Bloco 7: Alocado para 'projeto.c'  
Bloco 8: Alocado para 'atividade.py'  
Bloco 9: Livre  
Bloco 10: Livre  
Bloco 11: Livre  
Bloco 12: Livre
```

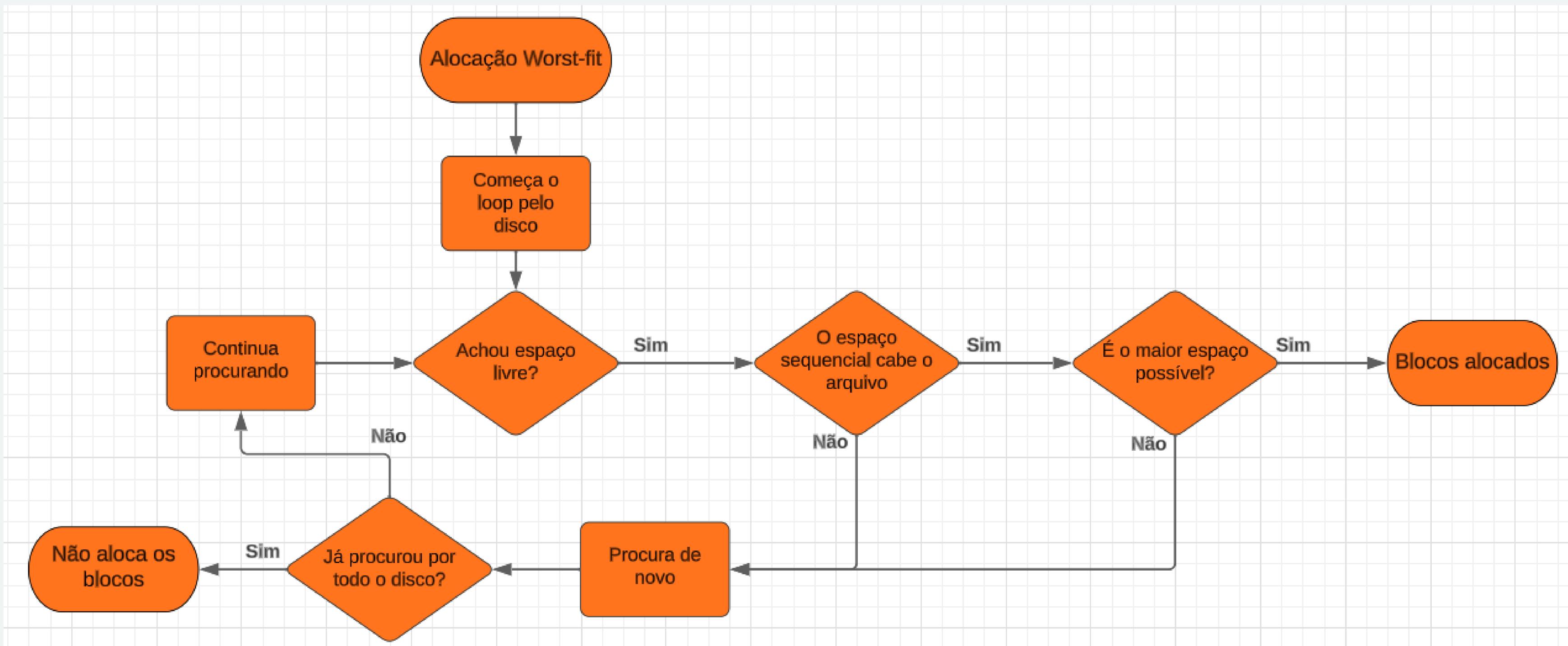
Input do usuário:

```
create programa.py 1 best-fit
```

Situação final do disco:

```
Bloco 0: Alocado para 'nota.txt'  
Bloco 1: Livre  
Bloco 2: Livre  
Bloco 3: Livre  
Bloco 4: Alocado para 'foto.png'  
Bloco 5: Alocado para 'programa.py'  
Bloco 6: Livre  
Bloco 7: Alocado para 'projeto.c'  
Bloco 8: Alocado para 'atividade.py'  
Bloco 9: Livre  
Bloco 10: Livre  
Bloco 11: Livre  
Bloco 12: Livre
```

ALOCAÇÃO CONTÍGUA - WORST-FIT



ALOCAÇÃO CONTIGUA - WORST-FIT

Situação inicial do disco:

```
Bloco 0: Alocado para 'nota.txt'  
Bloco 1: Livre  
Bloco 2: Livre  
Bloco 3: Livre  
Bloco 4: Alocado para 'foto.png'  
Bloco 5: Livre  
Bloco 6: Livre  
Bloco 7: Alocado para 'projeto.c'  
Bloco 8: Alocado para 'atividade.py'  
Bloco 9: Livre  
Bloco 10: Livre  
Bloco 11: Livre  
Bloco 12: Livre
```

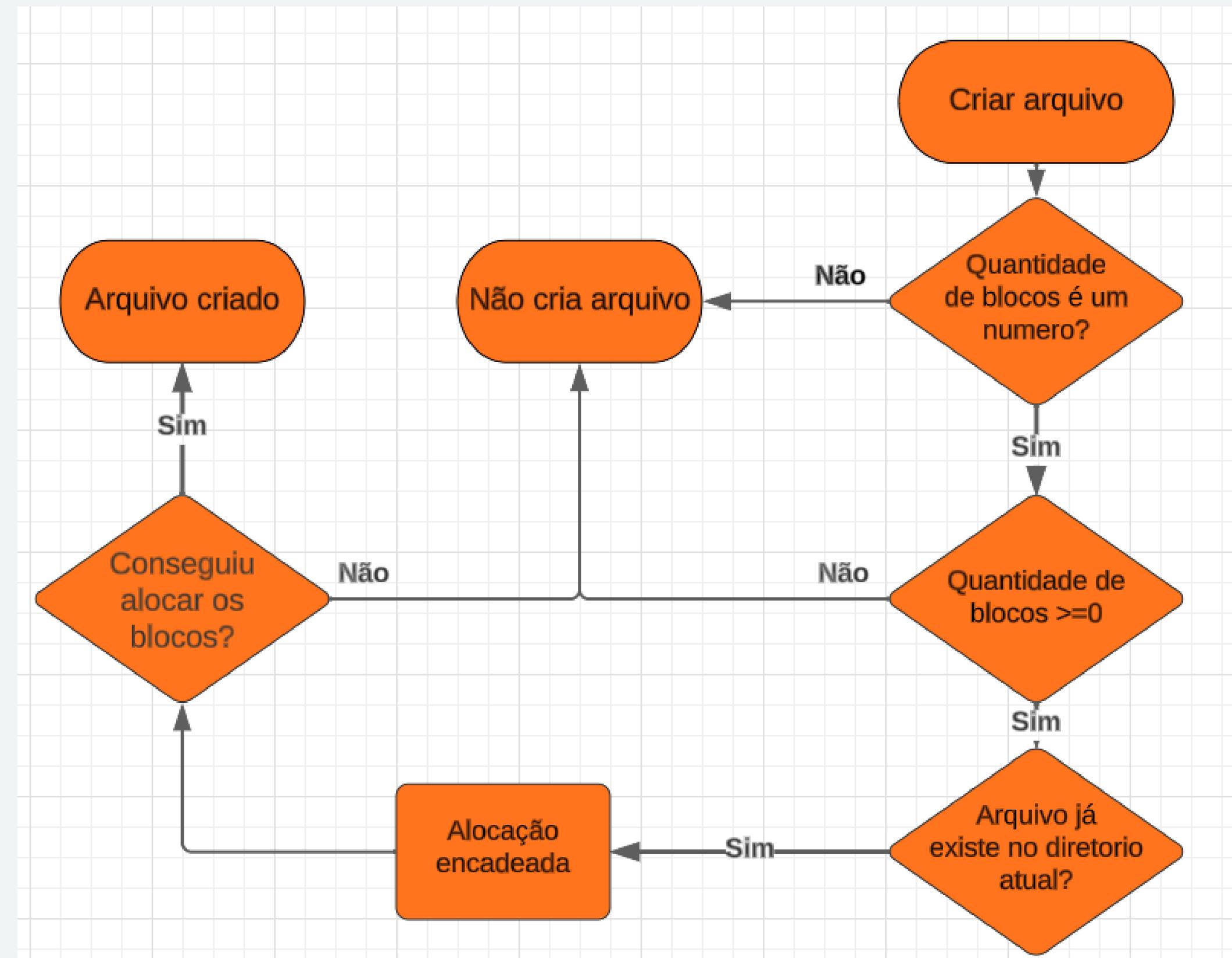
Input do usuário:

```
create programa.py 1 worst-fit
```

Situação final do disco:

```
Bloco 0: Alocado para 'nota.txt'  
Bloco 1: Livre  
Bloco 2: Livre  
Bloco 3: Livre  
Bloco 4: Alocado para 'foto.png'  
Bloco 5: Livre  
Bloco 6: Livre  
Bloco 7: Alocado para 'projeto.c'  
Bloco 8: Alocado para 'atividade.py'  
Bloco 9: Alocado para 'programa.py'  
Bloco 10: Livre  
Bloco 11: Livre  
Bloco 12: Livre
```

CRIAR ARQUIVOS - ALOCAÇÃO ENCADEADA



CRIAR ARQUIVO - ALOCAÇÃO ENCADEADA

Para a criação de um arquivo no programa é preciso o usuário digitar “create” e separar por espaços os parâmetros:

- Nome do arquivo.
- Quantidades de blocos a serem alocados.

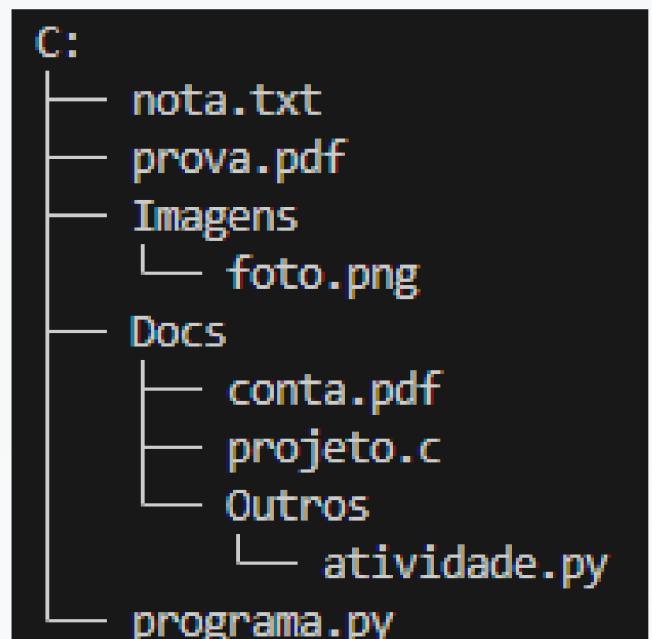
Exemplo de input do usuário:

```
create programa.py 3
```

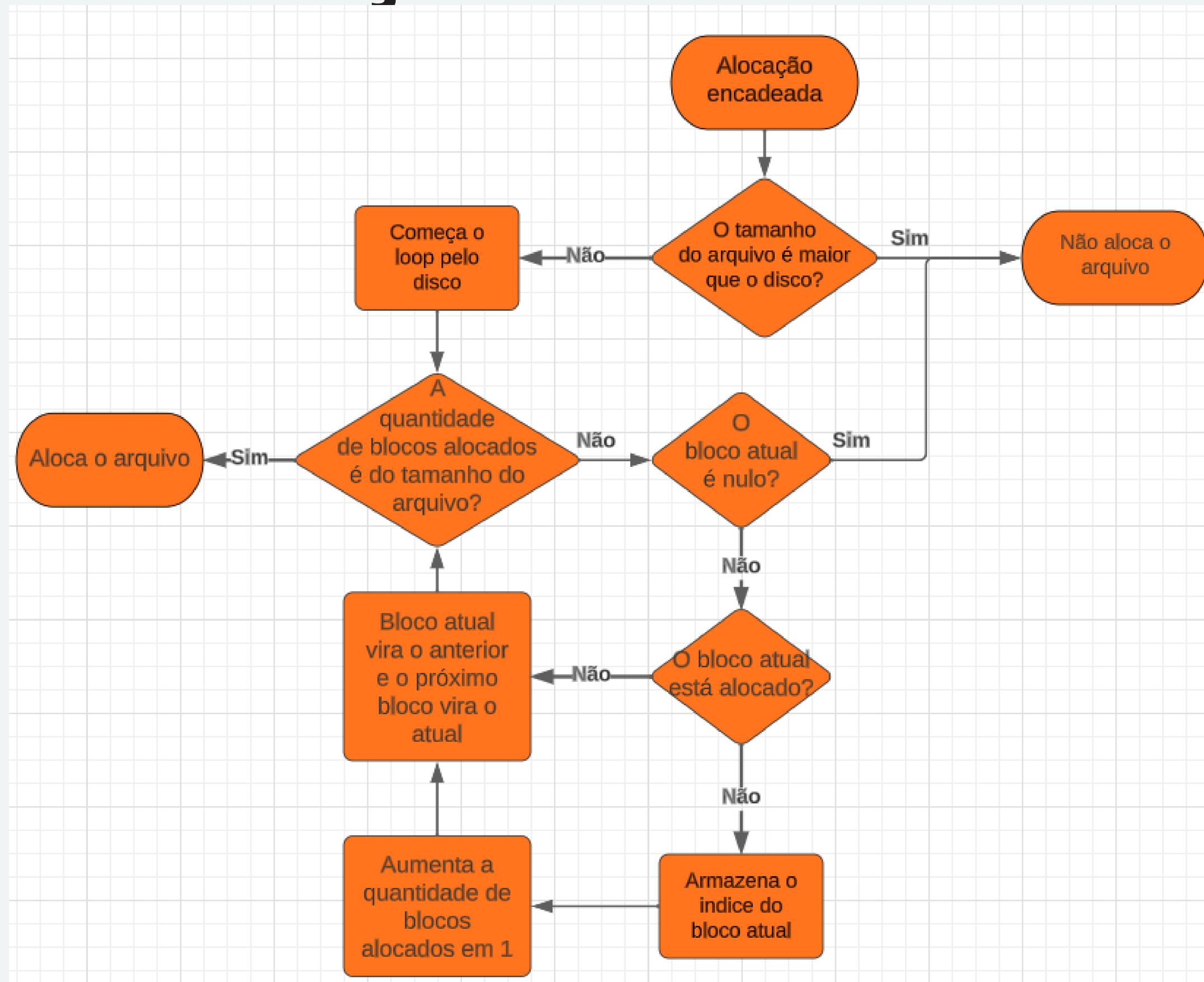
Exemplo de output do sistema:

```
Arquivo 'programa.py' criado com sucesso.
```

Como o arquivo é exibido no sistema de árvore:



ALOCAÇÃO ENCADEADA



ALOCAÇÃO ENCADEADA

Situação inicial do disco:

```
Bloco 0: Livre
Bloco 1: Livre
Bloco 2: Livre
Bloco 3: Alocado
Bloco 4: Alocado
Bloco 5: Alocado
Bloco 6: Alocado
Bloco 7: Alocado
Bloco 8: Alocado
Bloco 9: Alocado
Bloco 10: Livre
Bloco 11: Livre
Bloco 12: Livre
-----
Bloco 3: Alocado para 'notas.txt'
Bloco 4: Alocado para 'notas.txt'
Bloco 5: Alocado para 'notas.txt'
Bloco 6: Alocado para 'notas.txt'
Bloco 7: Alocado para 'notas.txt'
Bloco 8: Alocado para 'notas.txt'
Bloco 9: Alocado para 'notas.txt'
```

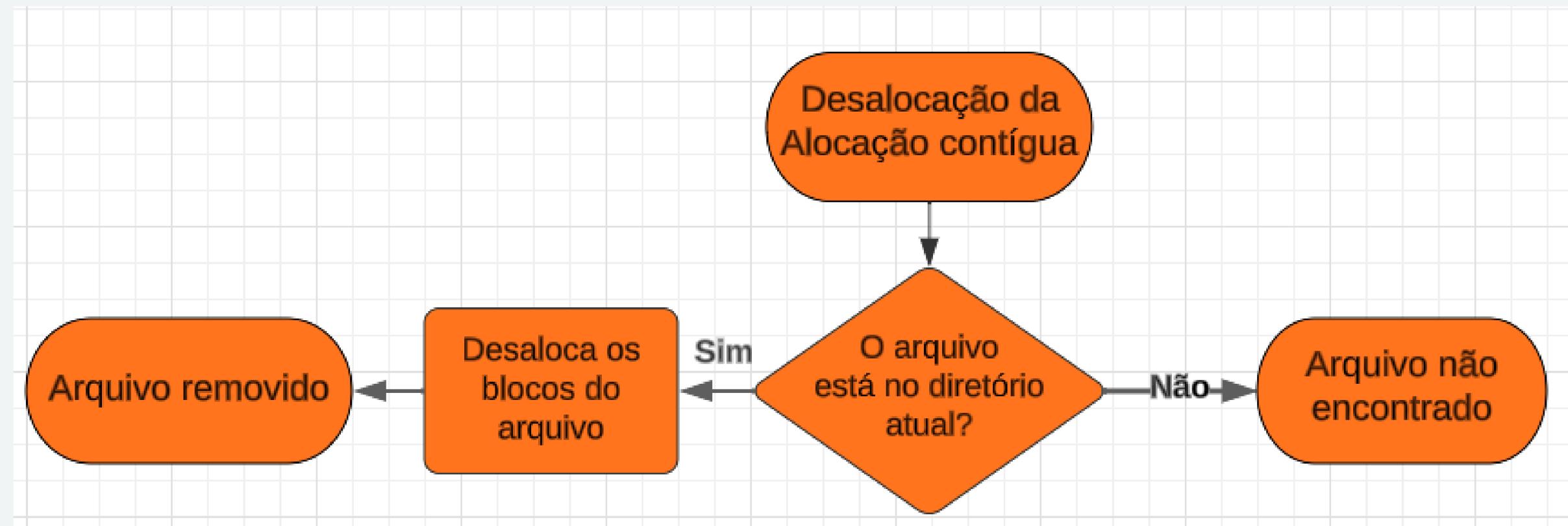
Input do usuário:

```
create programa.py 5
```

Situação final do disco:

```
Bloco 0: Alocado
Bloco 1: Alocado
Bloco 2: Alocado
Bloco 3: Alocado
Bloco 4: Alocado
Bloco 5: Alocado
Bloco 6: Alocado
Bloco 7: Alocado
Bloco 8: Alocado
Bloco 9: Alocado
Bloco 10: Alocado
Bloco 11: Alocado
Bloco 12: Livre
-----
Bloco 3: Alocado para 'notas.txt'
Bloco 4: Alocado para 'notas.txt'
Bloco 5: Alocado para 'notas.txt'
Bloco 6: Alocado para 'notas.txt'
Bloco 7: Alocado para 'notas.txt'
Bloco 8: Alocado para 'notas.txt'
Bloco 9: Alocado para 'notas.txt'
Bloco 0: Alocado para 'programa.py'
Bloco 1: Alocado para 'programa.py'
Bloco 2: Alocado para 'programa.py'
Bloco 10: Alocado para 'programa.py'
Bloco 11: Alocado para 'programa.py'
```

REMOVER ARQUIVO



REMOVER ARQUIVO

Para remover um arquivo no sistema é preciso o usuário digitar “remove” e separar por espaços o parâmetro:

- Nome do arquivo.

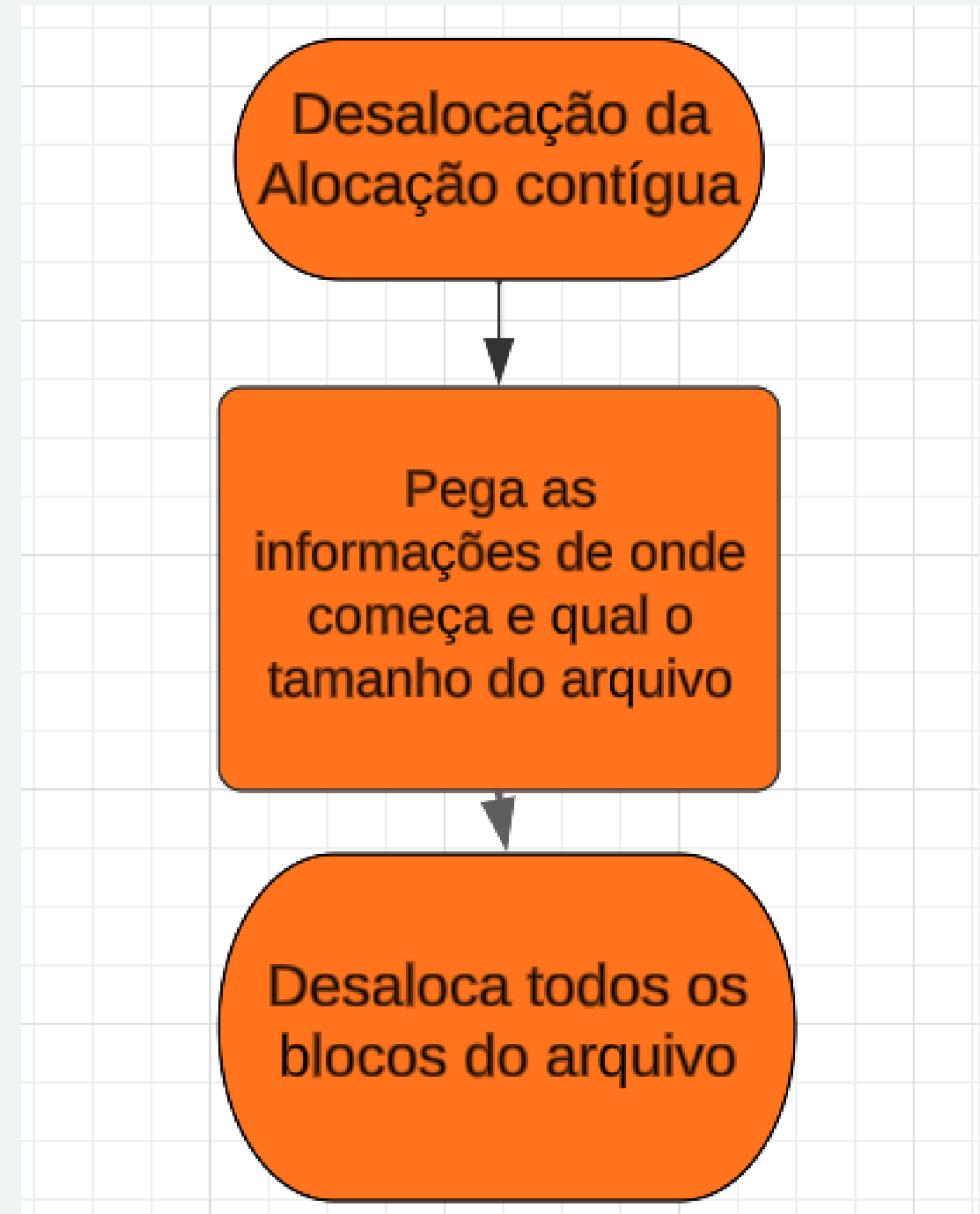
Exemplo de input do usuário:

```
SimpleOS> remove ArquivoTeste
```

Exemplo de output do sistema:

```
Arquivo 'ArquivoTeste' removido com sucesso e blocos desalocados.
```

REMOÇÃO DE BLOCOS - ALOCAÇÃO CONTÍGUA



REMOÇÃO DE BLOCOS - ALOCAÇÃO CONTÍGUA

Situação inicial do disco:

```
Bloco 0: Alocado para 'nota.txt'  
Bloco 1: Alocado para 'prova.pdf'  
Bloco 2: Alocado para 'prova.pdf'  
Bloco 3: Alocado para 'prova.pdf'  
Bloco 4: Alocado para 'foto.png'  
Bloco 5: Alocado para 'conta.pdf'  
Bloco 6: Alocado para 'conta.pdf'  
Bloco 7: Alocado para 'projeto.c'  
Bloco 8: Alocado para 'atividade.py'  
Bloco 9: Livre  
Bloco 10: Livre  
Bloco 11: Livre  
Bloco 12: Livre
```

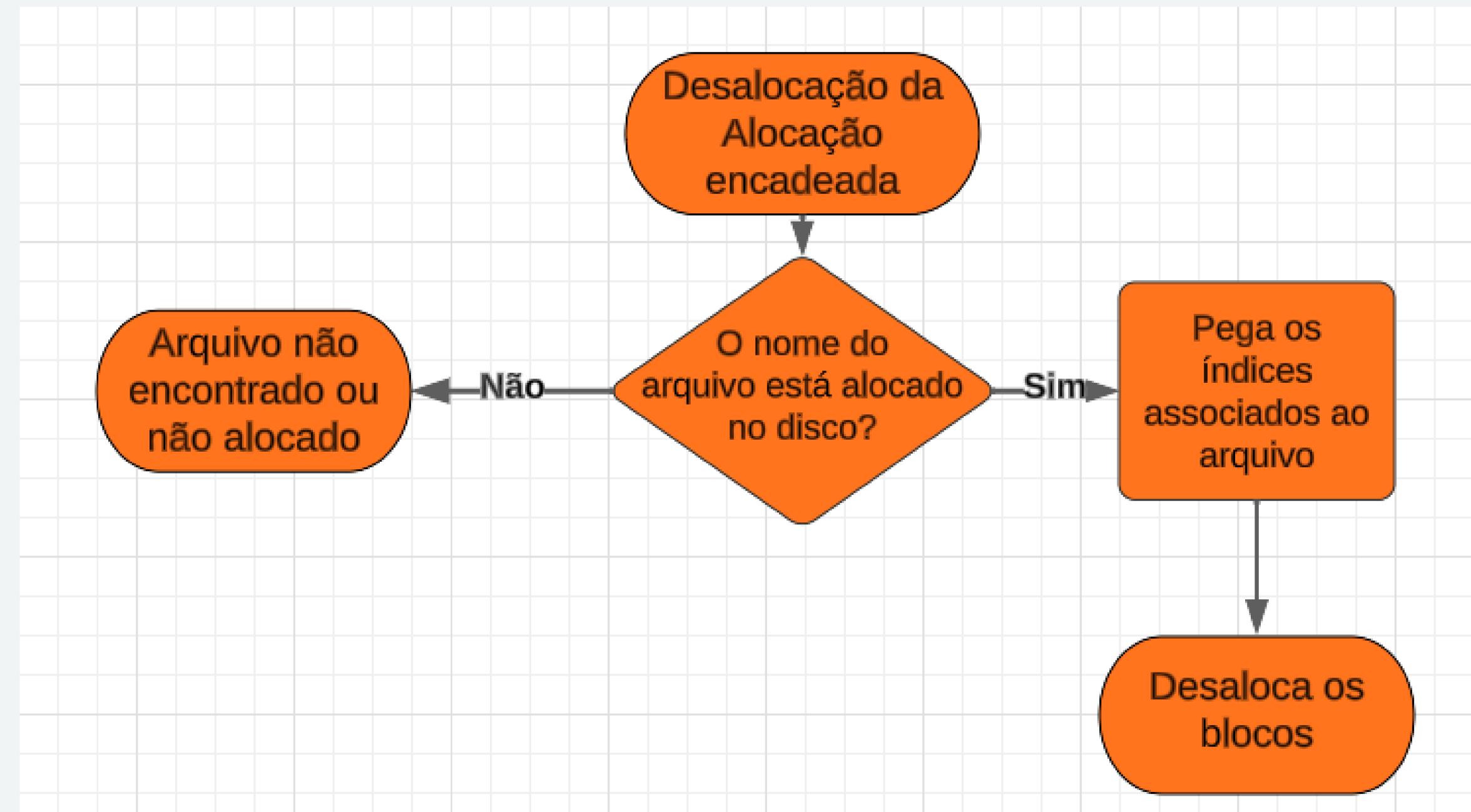
Input do usuário:

```
remove prova.pdf
```

Situação final do disco:

```
Bloco 0: Alocado para 'nota.txt'  
Bloco 1: Livre  
Bloco 2: Livre  
Bloco 3: Livre  
Bloco 4: Alocado para 'foto.png'  
Bloco 5: Alocado para 'conta.pdf'  
Bloco 6: Alocado para 'conta.pdf'  
Bloco 7: Alocado para 'projeto.c'  
Bloco 8: Alocado para 'atividade.py'  
Bloco 9: Livre  
Bloco 10: Livre  
Bloco 11: Livre  
Bloco 12: Livre
```

REMOÇÃO DE BLOCOS - ALOCAÇÃO ENCADEADA



ALOCAÇÃO ENCADEADA - REMOÇÃO DE BLOCOS

Situação inicial do disco:

```
Bloco 0: Alocado
Bloco 1: Alocado
Bloco 2: Alocado
Bloco 3: Alocado
Bloco 4: Alocado
Bloco 5: Alocado
Bloco 6: Alocado
Bloco 7: Alocado
Bloco 8: Alocado
Bloco 9: Alocado
Bloco 10: Alocado
Bloco 11: Alocado
Bloco 12: Livre
-----
Bloco 3: Alocado para 'prova.pdf'
Bloco 4: Alocado para 'prova.pdf'
Bloco 5: Alocado para 'prova.pdf'
Bloco 6: Alocado para 'prova.pdf'
Bloco 7: Alocado para 'prova.pdf'
Bloco 8: Alocado para 'prova.pdf'
Bloco 9: Alocado para 'prova.pdf'
Bloco 0: Alocado para 'programa.py'
Bloco 1: Alocado para 'programa.py'
Bloco 2: Alocado para 'programa.py'
Bloco 10: Alocado para 'programa.py'
Bloco 11: Alocado para 'programa.py'
```

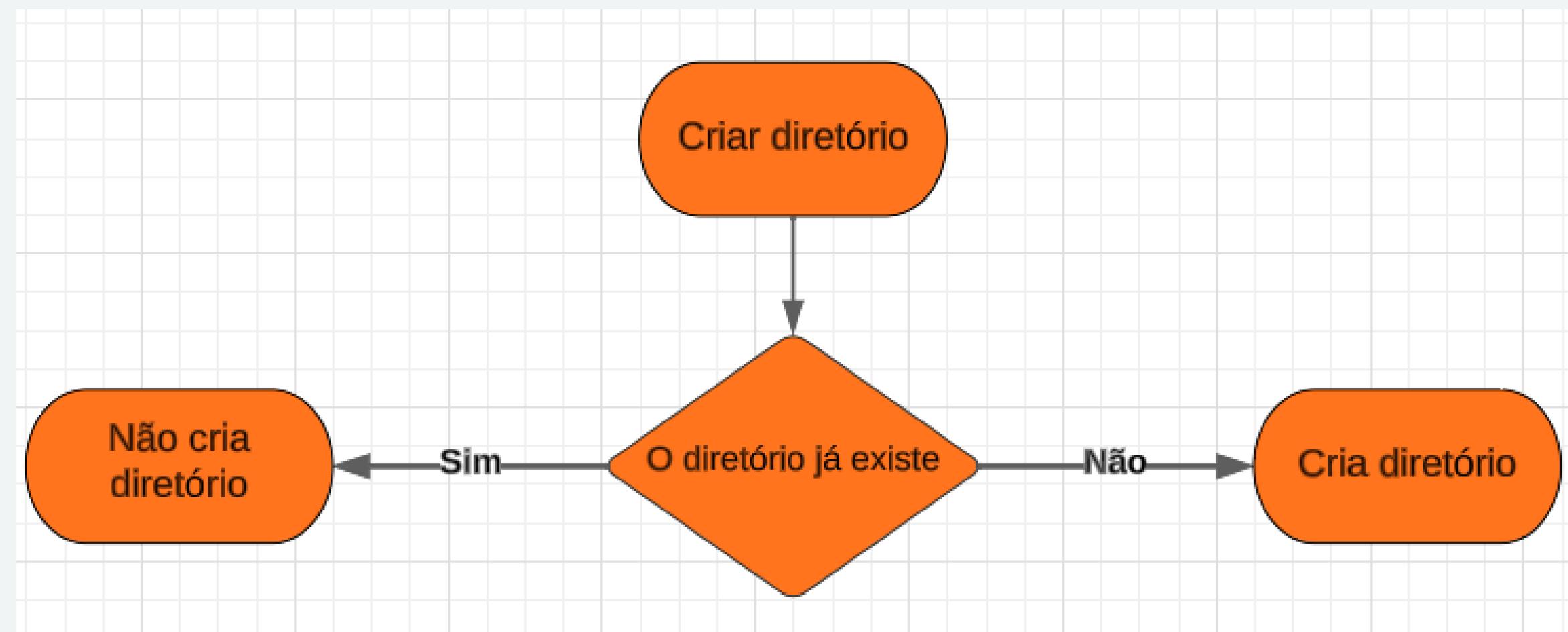
Input do usuário:

```
remove programa.py
```

Situação final do disco:

```
Bloco 0: Livre
Bloco 1: Livre
Bloco 2: Livre
Bloco 3: Alocado
Bloco 4: Alocado
Bloco 5: Alocado
Bloco 6: Alocado
Bloco 7: Alocado
Bloco 8: Alocado
Bloco 9: Alocado
Bloco 10: Livre
Bloco 11: Livre
Bloco 12: Livre
-----
Bloco 3: Alocado para 'prova.pdf'
Bloco 4: Alocado para 'prova.pdf'
Bloco 5: Alocado para 'prova.pdf'
Bloco 6: Alocado para 'prova.pdf'
Bloco 7: Alocado para 'prova.pdf'
Bloco 8: Alocado para 'prova.pdf'
Bloco 9: Alocado para 'prova.pdf'
```

FLUXOGRAMA - CRIAR DIRETÓRIO



CRIAR DIRETÓRIO

Para criar um diretório de arquivos no sistema é preciso o usuário digitar “mkdir” e separar por espaços o parâmetro:

- Nome do diretório.

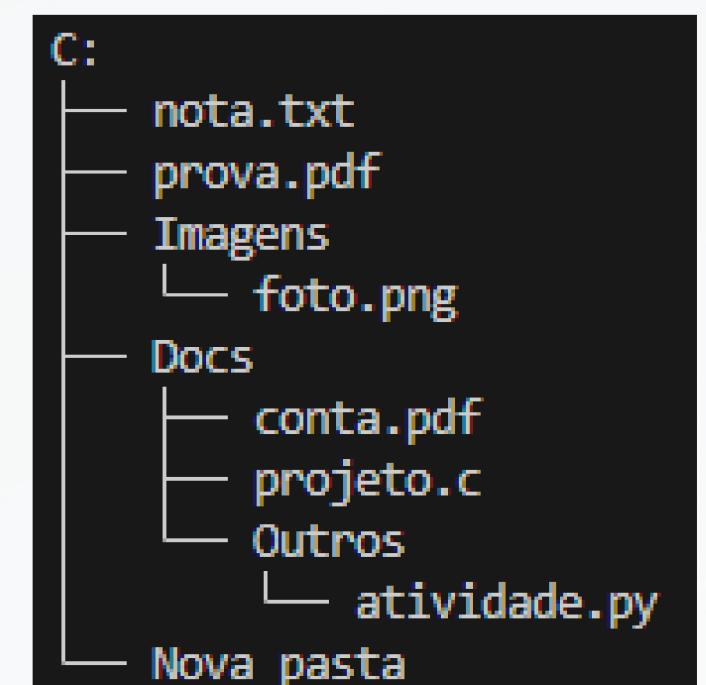
Exemplo de input do usuário:

```
mkdir Nova_pasta
```

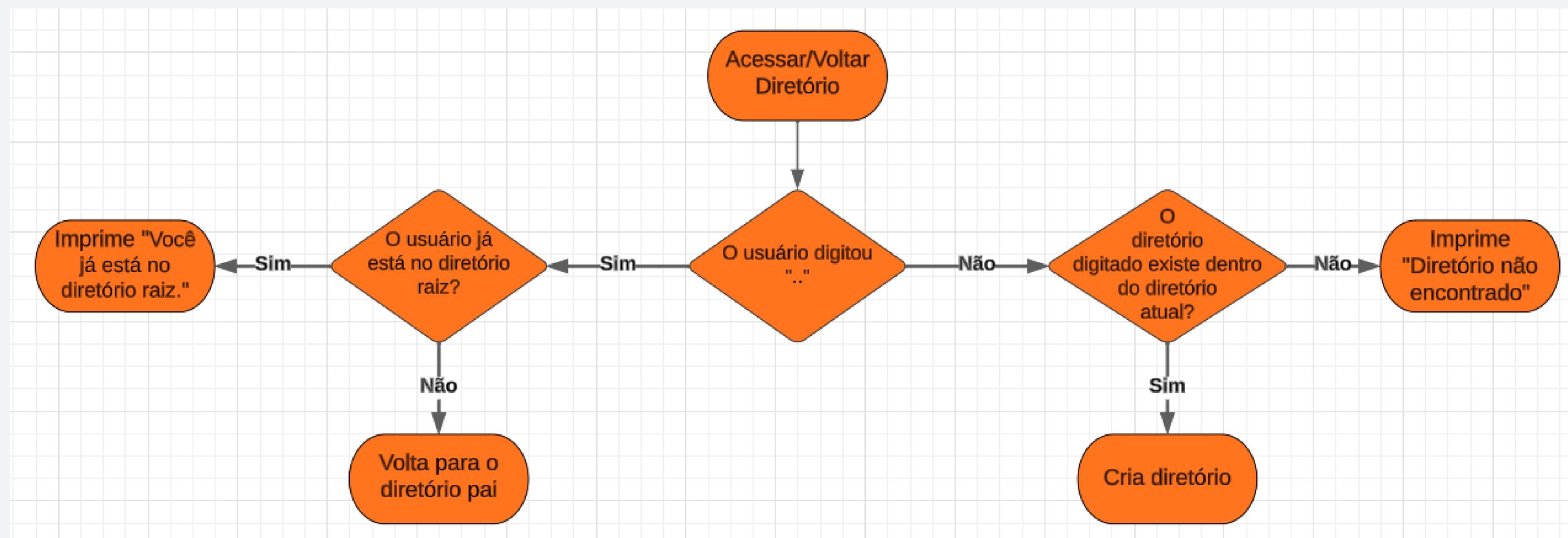
Exemplo de output do sistema:

```
Diretório 'Nova_pasta' criado com sucesso.
```

Como o arquivo é exibido no sistema de árvore:



FLUXOGRAMA - ACESSAR/VOLTAR DIRETÓRIO



ACESSAR/VOLTAR DIRETÓRIO

Para navegar no sistema é preciso o usuário digitar “cd” e separar por espaços o parâmetro:

- Nome do diretório para acessar.
- digitar “..” – para voltar ao diretório pai

Exemplo de input do usuário:

```
cd Docs
```

Exemplo de output do sistema:

```
Entrou no diretório 'Docs'.
```

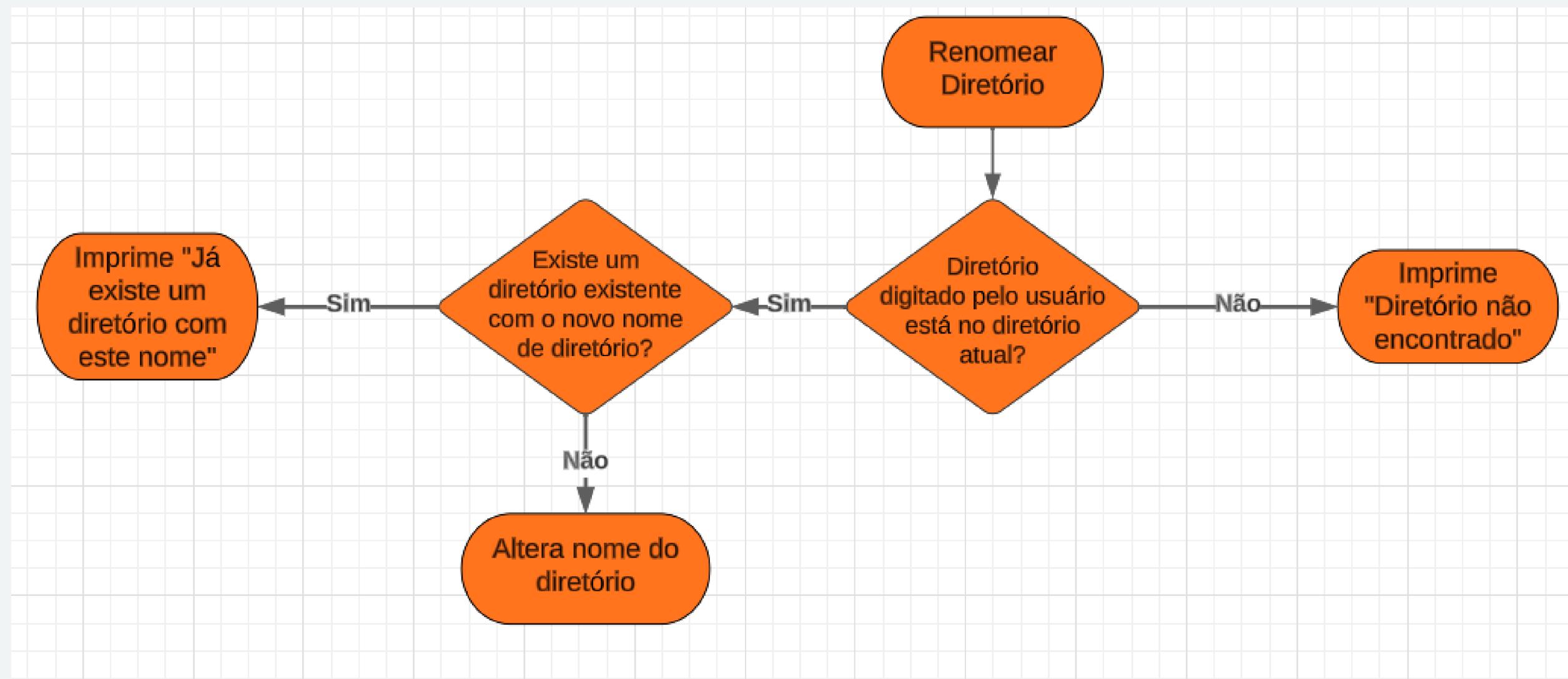
Situação inicial:

```
C:
  └── nota.txt
  └── prova.pdf
      └── Imagens
          └── foto.png
  └── Docs
      └── conta.pdf
      └── projeto.c
  └── Outros
      └── atividade.py
```

Situação final:

```
Diretório atual: C:/Docs
Docs
  └── conta.pdf
  └── projeto.c
  └── Outros
      └── atividade.py
```

FLUXOGRAMA - RENOMEAR DIRETÓRIO



RENOMEAR DIRETÓRIOS

Para renomear um diretório no sistema é preciso o usuário digitar “renamedir” e separar por espaços o parâmetro:

- Nome antigo do diretório.
- Nome novo do diretório

Exemplo de input do usuário:

```
SimpleOS> renamedir Imagens Fotos
```

Exemplo de output do sistema:

```
Diretório 'Imagens' renomeado para 'Fotos' com sucesso.
```

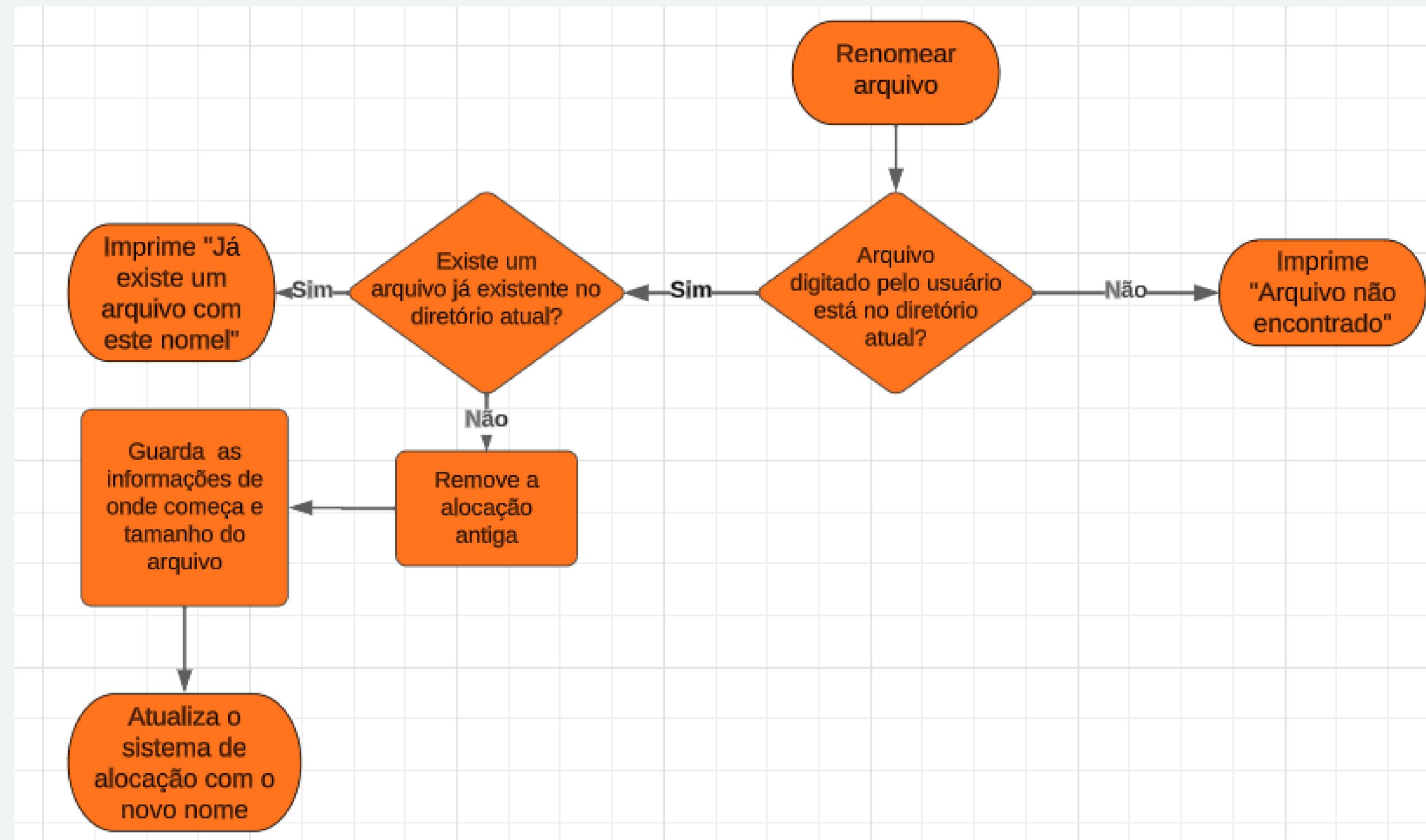
Situação inicial:

```
C:
  └── nota.txt
  └── prova.pdf
  └── Imagens
      └── foto.png
  └── Docs
      └── conta.pdf
      └── projeto.c
  └── Outros
      └── atividade.py
```

Situação final:

```
C:
  └── nota.txt
  └── prova.pdf
  └── Fotos
      └── foto.png
  └── Docs
      └── conta.pdf
      └── projeto.c
  └── Outros
      └── atividade.py
```

FLUXOGRAMA - RENOMEAR ARQUIVO



RENOMEAR ARQUIVO

Para remover um arquivo no sistema é preciso o usuário digitar “remove” e separar por espaços o parâmetro:

- Nome do arquivo.

Exemplo de input do usuário:

```
rename prova.pdf PROVA.pdf
```

Exemplo de output do sistema:

```
Arquivo 'prova.pdf' renomeado para 'PROVA.pdf' com sucesso.
```

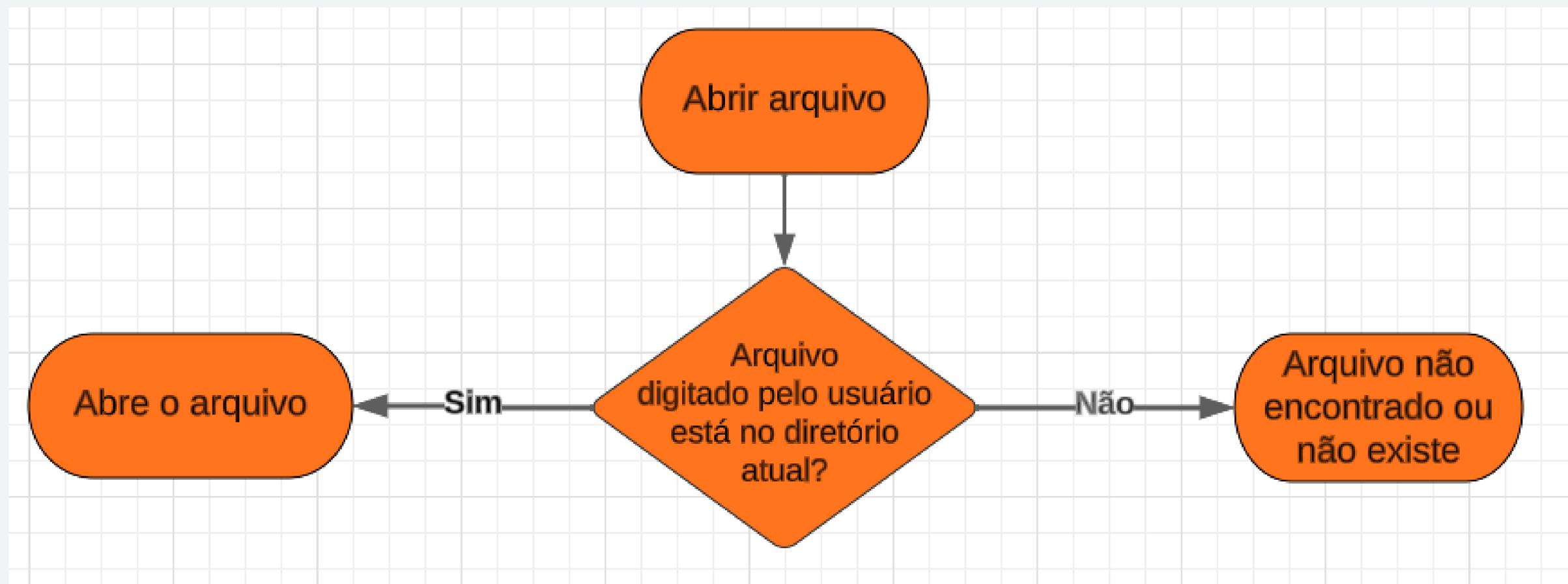
Situação inicial:

```
C:
 └── nota.txt
 └── prova.pdf
   └── Imagens
     └── foto.png
   └── Docs
     └── conta.pdf
     └── projeto.c
   └── Outros
     └── atividade.py
```

Situação final:

```
C:
 └── nota.txt
 └── PROVA.pdf
   └── Imagens
     └── foto.png
   └── Docs
     └── conta.pdf
     └── projeto.c
   └── Outros
     └── atividade.py
```

FLUXOGRAMA - ABRIR ARQUIVO



ABRIR ARQUIVO

Para abrir um arquivo no sistema é preciso o usuário digitar “open” e separar por espaços o parâmetro:

- Nome do arquivo.

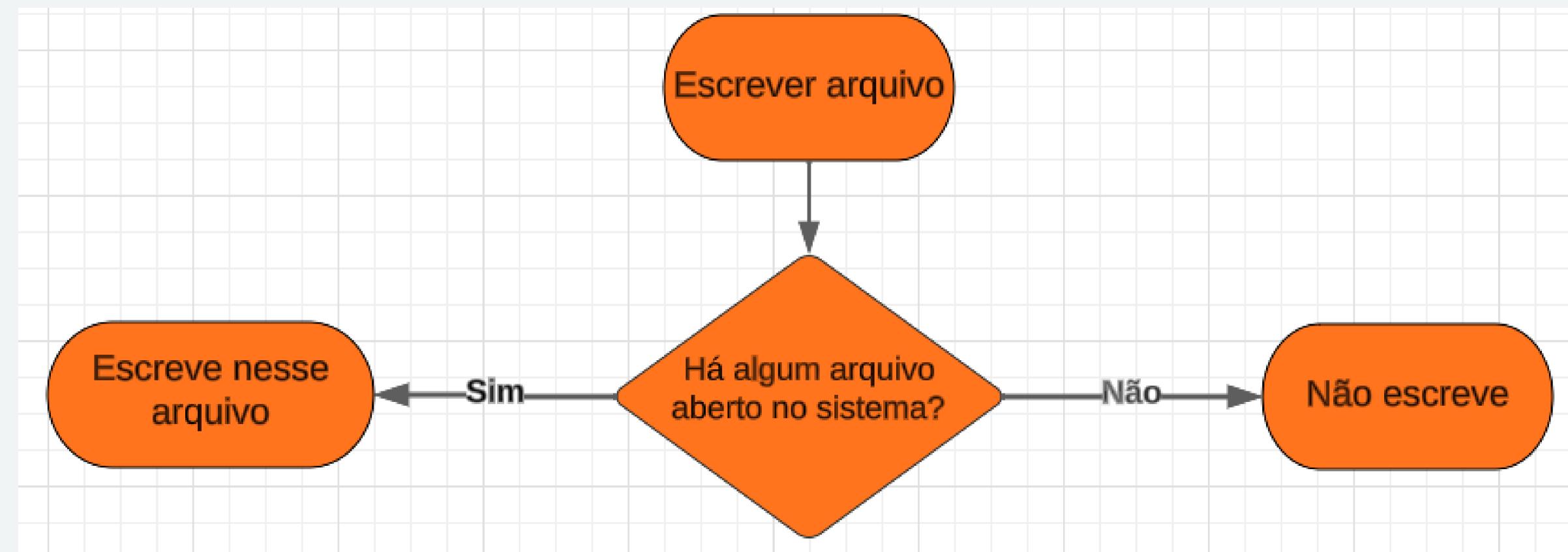
Exemplo de input do usuário:

```
SimpleOS> open programa.py
```

Exemplo de output do sistema:

```
Arquivo 'programa.py' aberto.
```

FLUXOGRAMA - ESCREVER ARQUIVO



ESCREVER ARQUIVO

Para escrever um conteúdo no arquivo é necessário que o usuário digite “write” após abrir o arquivo e separar por espaços o parâmetro:

- Conteúdo a ser escrito no arquivo.

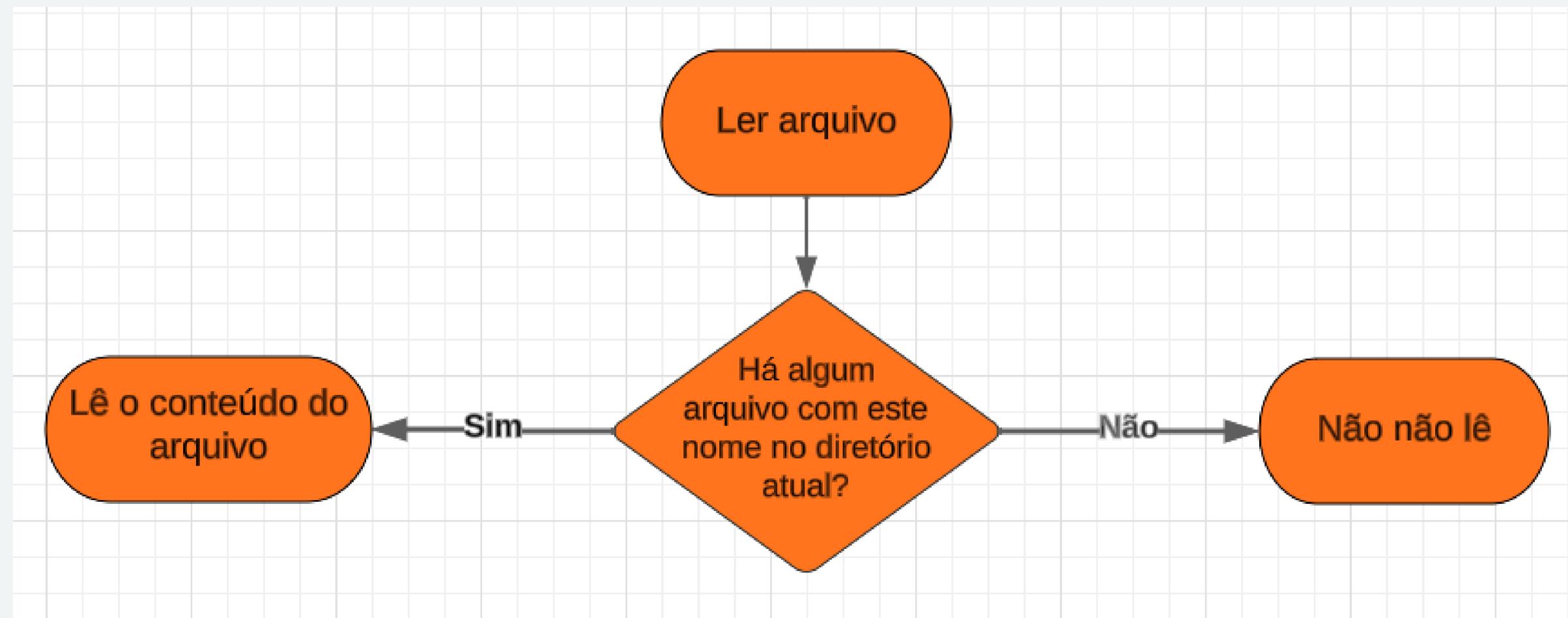
Exemplo de input do usuário:

```
SimpleOS> write Hello World
```

Exemplo de output do sistema:

```
Conteúdo escrito com sucesso.
```

FLUXOGRAMA - LER ARQUIVO



LER ARQUIVO

Para ler o conteúdo presente em um arquivo é preciso que o usuário digite “read” e separar por espaços o parâmetro:

- Nome do arquivo.

Exemplo de input do usuário:

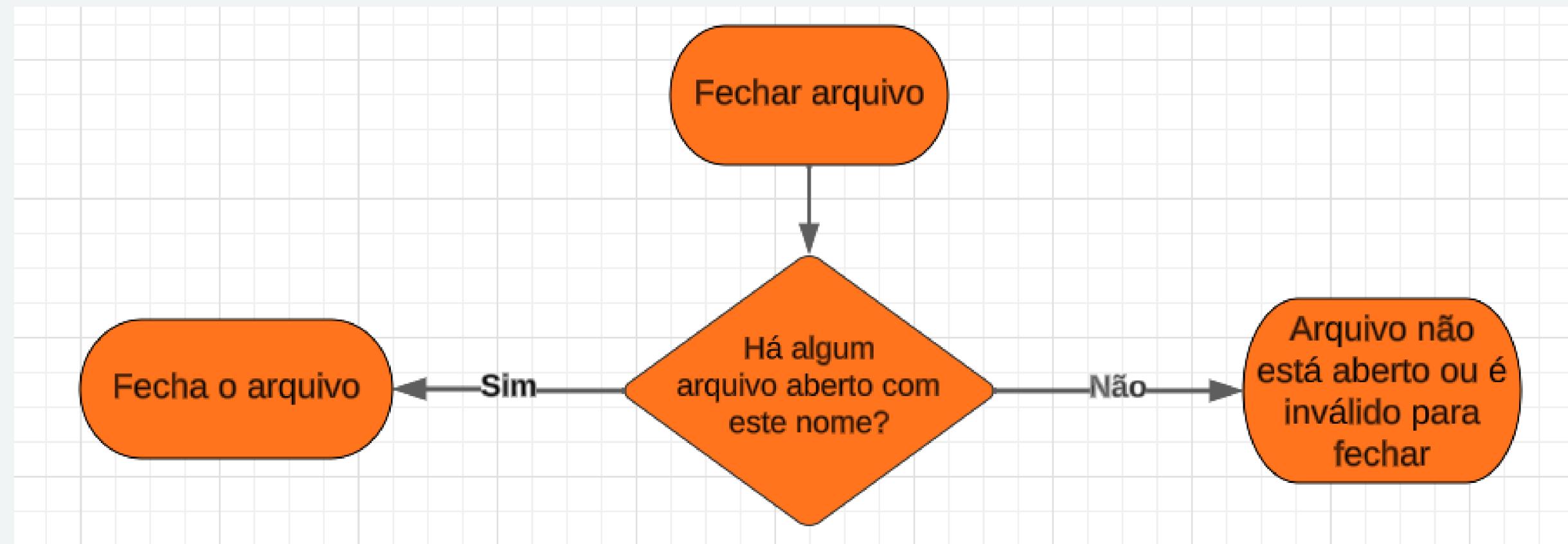
```
SimpleOS> read programa.py
```

Exemplo de output do sistema:

Conteúdo do arquivo:

```
Hello World
```

FLUXOGRAMA - FECHAR ARQUIVO



FECHAR ARQUIVO

Para fechar um arquivo que foi aberto é preciso o usuário digitar “close” e separar por espaços o parâmetro:

- Nome do arquivo.

Exemplo de input do usuário:

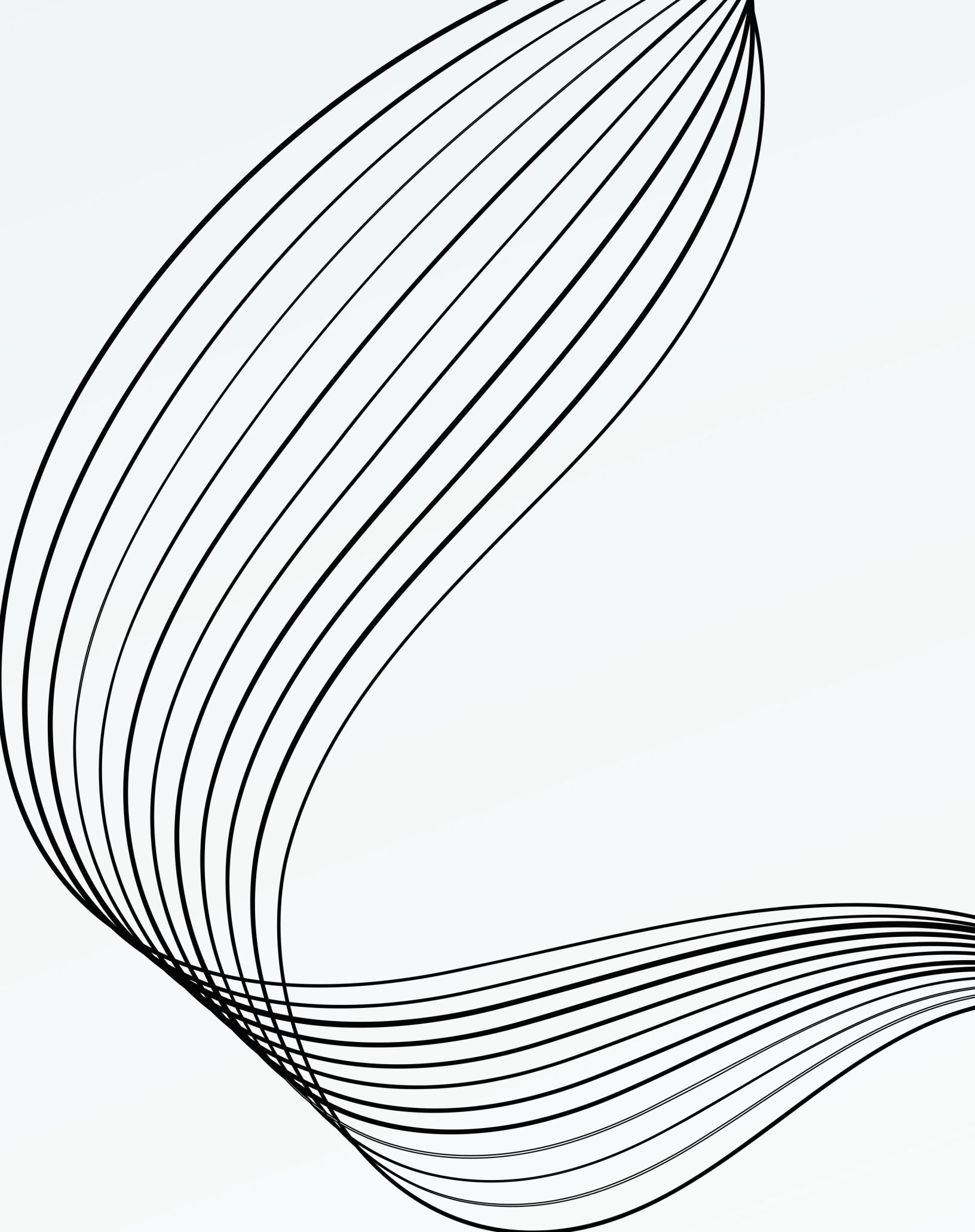
```
SimpleOS> close programa.py
```

Exemplo de output do sistema:

```
Arquivo 'programa.py' fechado com sucesso.
```

IMPLEMENTAÇÃO

Exibição do código



Reconhecimentos e Direitos Autorais

@autor: Gustavo de Oliveira Rego Moraes e Keven Gustavo dos Santos Gomes

@ contato: gustavo.moraes@discente.ufma.br e keven.gustavo@discente.ufma.br

@data última versão: 09/12/2023

@versão: 1.0

@outros repositórios: <https://github.com/gustvo-olive> e <https://github.com/KevenGustavo>

@Agradecimentos: Universidade Federal do Maranhão (UFMA), Professor Doutor Thales Levi Azevedo Valente, e colegas de curso.

@Copyright/License

Este material é resultado de um trabalho acadêmico para a disciplina **SISTEMAS OPERACIONAIS**, sobre a orientação do professor Dr. THALES LEVI AZEVEDO VALENTE, semestre letivo 2023.2, curso Engenharia da Computação, na Universidade Federal do Maranhão (UFMA). Todo o material sob esta licença é software livre: pode ser usado para fins acadêmicos e comerciais sem nenhum custo. Não há papelada, nem royalties, nem restrições de "copyleft" do tipo GNU. Ele é licenciado sob os termos da licença MIT reproduzida abaixo e, portanto, é compatível com GPL e também se qualifica como software de código aberto. É de domínio público. Os detalhes legais estão abaixo. O espírito desta licença é que você é livre para usar este material para qualquer finalidade, sem nenhum custo. O único requisito é que, se você usá-lo, nos dê crédito.

Copyright © 2023 Educational Material

Este material está licenciado sob a Licença MIT. É permitido o uso, cópia, modificação, e distribuição deste material para qualquer fim, desde que acompanhado deste aviso de direitos autorais.

O MATERIAL É FORNECIDO "COMO ESTÁ", SEM GARANTIA DE QUALQUER TIPO, EXPRESSA OU IMPLÍCITA, INCLUINDO, MAS NÃO SE LIMITANDO ÀS GARANTIAS DE COMERCIALIZAÇÃO, ADEQUAÇÃO A UM DETERMINADO FIM E NÃO VIOLAÇÃO. EM HIPÓTESE ALGUMA OS AUTORES OU DETENTORES DE DIREITOS AUTORAIS SERÃO RESPONSÁVEIS POR QUALQUER RECLAMAÇÃO, DANOS OU OUTRA RESPONSABILIDADE, SEJA EM UMA AÇÃO DE CONTRATO, ATO ILÍCITO OU DE OUTRA FORMA, DECORRENTE DE, OU EM CONEXÃO COM O MATERIAL OU O USO OU OUTRAS NEGOCIAÇÕES NO MATERIAL.

Para mais informações sobre a Licença MIT: <https://opensource.org/licenses/MIT>.