Editor de textos ASCII

Este projeto consiste em construir um editor de textos em terminal ASCII usando a biblioteca NCurses para gerenciar a interface com o usuário.

Funcionalidades

O programa deve implementar as seguintes funcionalidades:

- ler e escrever textos em arquivos de texto (`.txt`, `.c`, `.h`, etc)
- suportar textos de dimensões arbitrárias (milhares de linhas, centenas de colunas)
- navegar no texto usando as teclas de setas, *page up* e *page down*
- editar diretamente no texto, ao digitar teclas de caracteres
- implementar operações de cortar/copiar/colar usando um *clipboard*
- se adaptar às dimensões do terminal ao iniciar
- não precisa suportar acentos.

Controles

- Arquivos:
  - ^S : salvar arquivo corrente
  - ^R: ler novo arquivo
- Setas (←↑↓→) e *page up/down*: mover cursor
- Edição:
  - Enter: nova linha
  - DEL: deleta caractere sob o cursor
  - *backspace*: deleta caractere anterior
- Cortar/Copiar/colar:
  - ^space : marcar início do bloco de texto
  - ^C: copiar bloco de texto (entre marca e cursor) para o *clipboard*
  - ^X: recortar bloco de texto para o *clipboard*
  - ^V: colar *clipboard* na posição atual do cursor
- Outros:
  - ^F: busca uma string no texto
  - ^Q: encerrar o programa (*quit*)

As operações de ler novo arquivo ou encerrar o programa devem perguntar se o usuário deseja salvar o conteúdo atual antes!

Forma de chamada

- `mypad` : abre um texto em branco (sem nome)
- `mypad arq1.txt`: abre o arquivo `arq1.txt`

Requisitos do código-fonte

- Ser escrito em C padrão (C99 ou similar)
  - não gerar *warnings* ao usar flags `-Wall`
  - o executável deve se chamar `mypad` (de "*my Notepad*")
- o texto deve ser armazenado na memória usando alocação dinâmica (estruturas como listas encadeadas, ropes ou gap buffers podem ser usadas).
- o código-fonte deve ser estruturado em vários arquivos. Por exemplo:
  - `mypad.c` (programa principal)
  - `textfile.c`, `textfile.h` (lê/escreve arquivos TXT)
  - `clipboard.c`, `clipboard.h` (funções para manipular o *clipboard*)
  - ...
- Ter um Makefile:
  - alvos `all`, `clean` e `purge`
  - compilação e ligação devem ser separadas
- Ler e tratar estes arquivos de teste:
  - ... 

Estrutura do código

A estrutura geral de programas interativos, como editores, jogos, etc, usa a seguinte estrutura geral:

```
início
   inicializações

   repita
      desenha tela e posiciona cursor
      ler entrada do teclado
      processa entrada (atualiza estado interno)
   até fim
   finalizações
fim
```

Mais informações a respeito dessa estrutura podem ser obtidas em:

- https://web.archive.org/web/20161213145118/http://www.kathekonta.com:80/rlguide/
- https://gameprogrammingpatterns.com/game-loop.html
- https://brennan.io/2015/06/12/tetris-reimplementation/
- https://solarianprogrammer.com/2012/07/12/roguelike-game-cpp-11-part-1/

Material de apoio

Ncurses:

- Ncurses Programming Howto (leia **com atenção** a seção sobre inicialização)
- Writing Programs with Ncurses
- Ncurses Programming Guide
- Exemplos do ncurses no Linux, em `/usr/lib/ncurses/examples/` (pacote `ncurses-examples`)
- Código-fonte do pacote `ncurses-examples` disponível no Debian e derivados (Ubuntu, Mint, etc).

Instalação da biblioteca NCurses no Linux (Ubuntu, Debian, Mint):

```
sudo apt-get install libncurses-dev
```

O código de inicialização da biblioteca NCurses, com as configurações necessárias para este projeto, é o seguinte:

```
  initscr () ;       // inicializa a biblioteca ncurses
  raw () ;       // permite tratar teclas de controle (^C, ^Z, etc)
  keypad (stdscr, TRUE) ; // habilita leitura de teclas de setas, Fn, etc
  noecho () ;        // não escreve as teclas lidas na tela
  curs_set (0) ;    // esconde o cursor do terminal
```