Для редактирования файлов в консоли предлагается два основных варианта - nano и vim
nano - примитивный, скучный и глупый (но осваивается быстро)
vim - изначально кажется сложным и неудобным, но при настройке и появлении привычки становится мощным и удобным инструментом

Зачем это всё, если есть среда (типа codeblocks)?
Оказавшись в ситуации без среды, к которой привык (к примеру, комиссия по праку у Чернова или если графическая оболочка linux полетит, что возможно и нередко), будешь тупить и долго разбираться/вспоминать, что к чему
vim можно запустить на компе без графической оболочки, НО в отличие, к примеру, от nano, он может предоставить все её возможности в консоли, если (!) его правильно настроить (что на линуксе, кстати, возможно без прав суперюзера)
к тому же vim есть для всех платформ (в том числе и для windows)

Что делать в vim:
открыть файл командой
vim <название>
если команда не работает, нужно установить стандартной командой sudo apt-get install vim

после команды откроется программа
чтобы редактировать файл, нажать клавишу i
(снизу появится insert - режим)
чтобы выйти из режима insert, нажать esc
надпись снизу пропадёт
чтобы выйти из редактора в консоль, выйти из режима редактирования и написать :q
(текст команды, начинающейся на ":", появляется снизу)
но так он не выйдет, если есть несохранённые изменения
если файл не был сохранён, нужно ввести :w, чтобы сохранить, :wq, чтобы сохранить и выйти сразу, или :q!, чтобы обойти защиту и выйти без сохранения

Настройка vim: (удобная для прака)

изначально vim весьма неудобен, это можно исправить
для этого нужно перейти в домашний каталог (командой cd без параметров)
открыть/создать файл ".vimrc" командой "vim .vimrc"
нажать i (режим insert)
вписать туда 6 строчек:

```
set tabstop=4
set shiftwidth=4
set smarttab
set expandtab
set mouse=a
set smartindent
```

нажать esc (выход из insert)
написать :wq, нажать enter (сохранить файл и выйти)

"set tabstop=4"
— длина отображения tab в тексте
"set expandtab"
— в режиме вставки заменяет символ табуляции на соответствующее количество пробелов
"set shiftwidth"
— по умолчанию используется для регулирование ширины отступов в пробелах, добавляемых командами » и «. Если значение опции не равно tabstop, как и в случае с softtabstop, отступ может состоять как из символов табуляций так и из пробелов. При включении опции — smarttab, оказывает дополнительное влияние.
"set smarttab"
— в случае включения этой опции, нажатие Tab в начале строки (если быть точнее, до первого непробельного символа в строке) приведет к добавлению отступа, ширина которого соответствует shiftwidth (независимо от значений в tabstop и softtabstop). Нажатие на Backspace удалит отступ, а не только один символ, что очень полезно при включенной expandtab. Напомню: опция оказывает влияние только на отступы в начале строки, в остальных местах используются значения из tabstop и softtabstop.
"set mouse=a"
— разрешает работать мышкой в редакторе, сдвигать курсор, выделять текст
"set smartindent"
autoindent — копирует отступы с текущей строки при добавлении новой (она не нужна)
smartindent — делает то же, что и autoindent плюс автоматически выставляет отступы в «нужных» местах. В частности, отступ ставится после строки, которая заканчивается символом {, перед строкой, которая заканчивается символом }, удаляется перед символом #, если он следует первым в строке и т.д. (подробнее help 'smartindent').
По-сути, после этих команд vim будет работать как полноценная среда, только в консоли

этот файлик с настройками .vimrc - локальный, не нужно быть суперпользователем, чтобы что-то настроить

Фишки vim

у vim не просто так отдельный режим редактирования insert
есть режим replace (замена), клавиша r
выйдя из режима, можно быстро перемещаться по коду (полезно для больших файлов)
":3" переведёт тебя на третью строку кода, ":5" на пятую, ":17" на 17-ую
вне режима вставки можно навести на строку, нажать dd (клавишу d два раза), чтобы вырезать строку, с помощью p вставить. А можно так просто убирать строки.

добавив в vimrc "set number", отобразятся номера строк
есть ещё функции автоподстановки команд c, c++, питона и многих других языков (нужно загуглить, как их включать); возможно компилировать прямо из редактора и сделать шаблоны компиляции; редактировать несколько файлов одновременно по вкладкам или в разделённом окне; включить сворачивание кода и т.д.

возможностей vim очень много, их имеет смысл поискать в интернете

Далее – небольшая справка/шпаргалка, взятая с сайта: https://eax.me/vim-commands/

Файл настройки: ~/.vimrc
Можно редактировать файлы через сеть, например
:e <scp|ftp|ftps>://user@host/path/to/the/file.txt
:Ex или :e ./ - файловый менеджер

== Основы ==

hjkl	перемещение в разные стороны
i	режим вставки
I	добавление в начало строки
a	режим добавления
A	добавление в конец строки
o	добавить строку сразу за текущей
O	добавить строку перед текущей
R	писать поверх имеющегося текста
u, :u[ndo]	отмена предыдущего действия (undo)
CTR-R, :red[o]	отмена отмены предыдущего действия (redo)
dd	вырезать (удалить) строку
cc	удалить и начать редактирование
yy	копировать строку
p	вставить из буфера обмена
<n>d	удалить n+1 строку
<n>y	скопировать n+1 строку
ESC	перейти в режим просмотра
DEL	удалить следующий символ
:<n>	перейти на строку #n
%	перейти к парной скобке
:e **/filename.c	редактировать файл (с поиском по имени)
:w [fname]	записать изменения
:wa	сохранить изменения во всех файлах
:q	выйти из редактора
:q!	выйти из редактора, не сохраняя изменения
:color <name>	выбор цветовой схемы. цветные схемы: /usr/local/share/vim/vim72/colors/*.vim
:pwd	текущий каталог
:cd [path]	перейти в другой каталог
:!команда	выполнить команду - man, git, и так далее
	стрелочками веерх и вниз можно автодополнять команды и искать по истории
CTR+p или CTR+n	автоматическое дополнение текста (в режиме редактирования)
CTR+r,=,<expr>	вставить выражение, например 5*2 - 3 (в режиме редактирования)
CTR+u, CTR+d	Page Up / Page Down
CTR+y, CTR+e	Перемотка вверх/вниз без движения курсора

== Подсветка синтаксиса ==
:syntax on включить подсветку
:syntax off выключить подсветку (по умолчанию)

== Перенос строк ==
:set wrap разрешить word wrap (по умолчанию)
:set nowrap запретить word wrap

== Печать ==
:ha[rdcopy] распечатать документ
:set printoptions=duplex:off отключить двустороннюю печать

== Сворачивание ==
zc свернуть блок
zo развернуть блок
zM закрыть все блоки
zR открыть все блоки
za инвертирование
zf см :set foldmethod=manual
:set foldenable включить сворачивание
:set foldmethod=syntax сворачивание на основе синтаксиса
:set foldmethod=indent сворачивание на основе отступов
:set foldmethod=manual выделяем участок с помощью v и говорим zf
:set foldmethod=marker сворачивание на основе маркеров в тексте
:set foldmarker=bigin,end задаем маркеры начала и конца блока

== Маркеры ==
ma установить локальный маркер a
mB установить глобальный маркер B
`c перейти к локальному маркеру c
`0 вернуться на позицию, на которой закончили
работу при закрытии vim
:marks просмотр маркеров
set viminfo='1000,f1 маркеры пишутся в ~/.viminfo, восстанавливаясь
при следующем запуске vim. маркер " хранит
последнюю позицию курсора в файле
== Сессии ==
mksession file.session сохранить текущую сессию
source file.session восстановить ранее сохраненную сессию

== Макросы ==
qa записать макрос с именем a
q в режиме записи макроса: закончить запись
@a выполнить макрос с именем a
@@ повторить последний макрос

== Регистры ==
"ayy скопировать строку в регистр a
"bdd вырезать строку и поместить в регистр b
"C2d вырезать три строки и дописать в конец
регистра C
:reg [name1][name2][...] просмотреть содержимое регистров

== Выделение ==
v + hjkl выделение текста
SHIFT + v выделить строку
CTR + v выделение прямоугольника

p вставить
y копировать
d удалить
gu к нижнему регистру
gU к верхнему регистру

== Отступы ==
[#]> сдвинуть выделенное вправо
[#]< сдвинуть выделенное влево
[#]>> сдвинуть строку вправо
[#]<< сдвинуть строку влево
set tabstop=# для табуляции используется # пробелов
set shiftwidth=# в командах отступа используется # пробелов
set [no]expandtab заменять ли табуляцию на соответствующее
число пробелов

== Поиск и замена в файле ==
/Выражение поиск выражения в файле
\cВыражение поиск без учета регистра
n следующее совпадение
N предыдущее совпадение
:%s/foo/bar/gi замена строк, см http://eax.me/regular-expr/

== Поиск по всему проекту ==
:vimgrep /EXPR/ **/*.c поиск по регулярному выражению
:copen показать все найденные места
:close скрыть все найденные места
:cn переход к следующему результату
:cp переход к предыдущему результату

== Нумерация строк ==
:set number включить нумерацию строк
:set nonumber отключить нумерацию строк

== Работа с вкладками (a.k.a табами) ==
:tabnew [fname] создать таб
:tabs вывести список табов
:tabn следующий таб
:tabp предыдущий таб
<n>gt перейти на таб #n
gt следующий таб
gT предыдущий таб
:tabm +1 переместить таб вперед на одну позицию
:tabm -1 переместить таб назад на одну позицию
:tabm 2 переместить таб на заданную позицию
(нумерация начинается с нуля)

== Работа с окнами ==
:split горизонтальное разбиение
:vsplit вертикальное разбиение
Ctr+W, затем
c закрыть окно
+- изменение высоты текущего окна
<> изменение ширины текущего окна
= установить равный размер окон
hjkl или стрелочки перемещение между окнами

== Проверка орфографии ==

```
mkdir -p ~/.vim/spell
cd ~/.vim/spell
wget http://ftp.vim.org/vim/runtime/spell/ru.koi8-r.sug
wget http://ftp.vim.org/vim/runtime/spell/ru.koi8-r.spl
wget http://ftp.vim.org/vim/runtime/spell/en.ascii.sug
wget http://ftp.vim.org/vim/runtime/spell/en.ascii.spl
```

```
:set spell spelllang=ru,en      включить проверку орфографии
:set nospell                    выключить проверку орфографии
]s                              следующее слово с ошибкой
[s                              предыдущее слово с ошибкой
z=                              замена слова на альтернативу из списка
zg                              good word
zw                              wrong word
zG                              ignore word
```

```
== Работа с кодировкой ==
e ++enc=<имя кодировки>        Редактирование файла в ??? кодировке
w ++enc=<имя кодировки>        Сохранить файл в новой кодировке
set fileencodings=utf-8,koi8-r  Список автоматически определяемых
                                кодировок в порядке убывания
                                приоритета
```

```
== Другое ==
:set [no]wildmenu         При авто-дополнении в командной строке над
                          ней выводятся возможные варианты
:set list                 Отображать табуляцию и переводы строк
q:                        История команд
.                         Повторение последней команды
```