create\_test\_files()

**Назначение**: Создает набор тестовых файлов с различными типами содержимого для проверки работы cat-подобной утилиты.

**Создаваемые файлы**:

1. 1.txt - базовый тестовый файл с обычным текстом, пустыми строками и строками с табуляцией
2. 2.txt - файл с множественными пустыми строками для проверки их обработки
3. 3.txt - содержит непечатаемые символы (0x01) и символ DEL (0x7F)
4. 4.txt - полностью пустой файл для проверки граничных случаев
5. 5.txt - файл с одной очень длинной строкой для проверки обработки длинных строк

run\_cmd(const char \*cmd)

**Назначение**: Выполняет команду в shell и возвращает ее вывод.

**Параметры**:

* cmd - строка с командой для выполнения в shell

**Возвращаемое значение**:

* Указатель на статический буфер с выводом команды или "CMD ERROR" при ошибке

**Особенности**:

* Использует статический буфер (не потокобезопасно)
* Читает вывод команды порциями по 256 байт
* Закрывает pipe после выполнения команды

test\_all\_combinations(bool verbose)

**Назначение**: Тестирует все возможные комбинации флагов с тестовыми файлами, сравнивая вывод system cat и s21\_cat.

**Параметры**:

* verbose - флаг подробного вывода (true - выводить все тесты, false - только ошибки)

**Алгоритм работы**:

1. Перебирает все возможные комбинации флагов (2^8 - 1 вариантов)
2. Для каждой комбинации:
   * Формирует команды для system cat и s21\_cat
   * Выполняет обе команды через run\_cmd()
   * Сравнивает вывод команд
3. Подсчитывает и выводит статистику по успешным/неуспешным тестам

main(int argc, char \*\*argv)

**Назначение**: Точка входа в программу, управляет процессом тестирования.

**Параметры**:

* argc - количество аргументов командной строки
* argv - массив аргументов командной строки

**Логика работы**:

1. Создает тестовые файлы (если их нет)
2. Определяет режим работы (обычный или подробный) на основе аргументов
3. Запускает тестирование через test\_all\_combinations()
4. Возвращает 0 при успешном завершении

Системные методы, используемые в коде

1. fopen() - открывает файл для чтения/записи
2. fprintf() - записывает форматированные данные в файл
3. fclose() - закрывает открытый файл
4. popen() - открывает процесс, создавая pipe для чтения/записи
5. fgets() - читает строку из файла/pipe
6. strcat() - объединяет строки
7. strcmp() - сравнивает две строки
8. snprintf() - форматирует строку с ограничением длины
9. pclose() - закрывает pipe, открытый popen()

Алгоритм работы программы

1. **Инициализация**:
   * Определяются константы (максимальное число флагов, размер буфера, тестовые файлы)
   * Создается массив тестируемых флагов cat
2. **Создание тестовых файлов** (create\_test\_files()):
   * Программа создает 5 тестовых файлов с разными характеристиками:
     + Обычный текст
     + Множественные пустые строки
     + Непечатаемые символы
     + Пустой файл
     + Длинные строки
3. **Тестирование комбинаций флагов** (test\_all\_combinations()):
   * Для каждой возможной комбинации флагов (всего 255 комбинаций):  
     a. Формируется строка с флагами  
     b. Создаются две команды - для system cat и s21\_cat  
     c. Команды выполняются через run\_cmd()  
     d. Результаты сравниваются  
     e. Ведутся счетчики успешных/неуспешных тестов
4. **Вывод результатов**:
   * В зависимости от режима (обычный/подробный) выводятся:
     + Либо только ошибки (обычный режим)
     + Либо все тесты с результатами (подробный режим)
   * В конце выводится общая статистика по тестам
5. **Завершение работы**:
   * Программа возвращает 0, указывая на успешное завершение

Программа позволяет проверить корректность реализации утилиты s21\_cat путем сравнения ее вывода с эталонной реализацией cat для всех возможных комбинаций флагов и различных типов входных файлов.