Лабораторная работа №5.

Работа в пакетном режиме и валидация входных данных.

1. Добавить в приложение командной строки возможность работать в пакетном режиме.

1.1. Изменить объявление функции `main`, добавив в него параметры для работы с данными командной строки `argc` и `argv`:

```
int main(int argc, char * argv[])
```

1.2. Удалить команды ввода значений из стандартного потока ввода `cin`.

1.3. Подключить файл заголовка стандартной библиотеки C:

```
#include <cstdlib>
```

1.4. Получить значения нижней и верхней границ диапазона из первого и второго параметров командной строки:

```
int a = atoi(argv[1]);
int b = atoi(argv[2]);
```

1.5. Проверить правильность работы приложения. Для этого ввести необходимые значения параметров запуска в поле "Параметры командной строки" настроек запуска на вкладке "Проекты", и выполнить запуск приложения.

1.6. Сохранить сделанные изменения в системе управления версиями.

2. Реализовать в приложении командной строки проверку правильности входных данных.

2.1. Добавить проверку на наличие необходимого количества параметров в командной строке. Параметры включают в себя имя самой команды, поэтому значение argc не должно быть менее 3:

```
if(argc < 3) {
     // вывести сообщение об ошибке и завершить работу программы
     std::cerr << "Missing parameters" << std::endl;
     return 1;
}
```

Обратите внимание, что сообщение об ошибке выводится в поток `cerr` (стандартный поток ошибок), а не `cout`. Параметр оператора `return`, отличный от 0, указывает операционной системе и другим программам, что при выполнении произошла ошибка. Для различных типов ошибок можно использовать разные числовые значения (коды ошибок).

2.2. Подключить заголовок модуля `sstream`, добавить потоки для чтения данных из строк параметров и команды чтения данных из этих потоков

```
istringstream astr(argv[1]), bstr(argv[2]);
int a, b;
astr >> a;
bstr >> b;
```

2.3. Добавить проверку того, что значение из `astr` прочитано правильно и полностью. Для этого можно использовать методы `fail` и `eof` объекта потока:

```
if(astr.fail() || !astr.eof()) {
     // вывести сообщение об ошибке и завершить работу программы
     std::cerr << "A must be an integer" << std::endl;
     return 2;
}
```

2.4. Аналогично 2.3. добавить проверку чтения значения `b`.

2.5. Проверить работу программы во всех возможных сценариях (недостаточно параметров, недопустимое значение `a`, недопустимое значение `b`).

2.6. Сохранить сделанные изменения в системе управления версиями.

3. Самостоятельно реализовать в приложении командной строки проверку корректности значений a и b (a > 0, b > 0, a < b). Сообщения выводить в стандартный поток ошибок. Для различных типов ошибок использовать разные коды завершения программы.

4. Самостоятельно реализовать аналогичную п.3 проверку корректности значений a и b в проекте графического приложения.