

Operating Systems. IHW 1

[Operating Systems](#)

Александр Васюков | БПИ235

Вариант 5

Разработать программу, заменяющую все строчные гласные буквы в заданной ASCII-строке заглавными.

Решение претендует на 10 баллов.

Программы можно найти на Github по ссылке: https://github.com/vasyukov1/HSE-FCS-SE-2-year/tree/main/Operating%20Systems/Homeworks/IHW_1

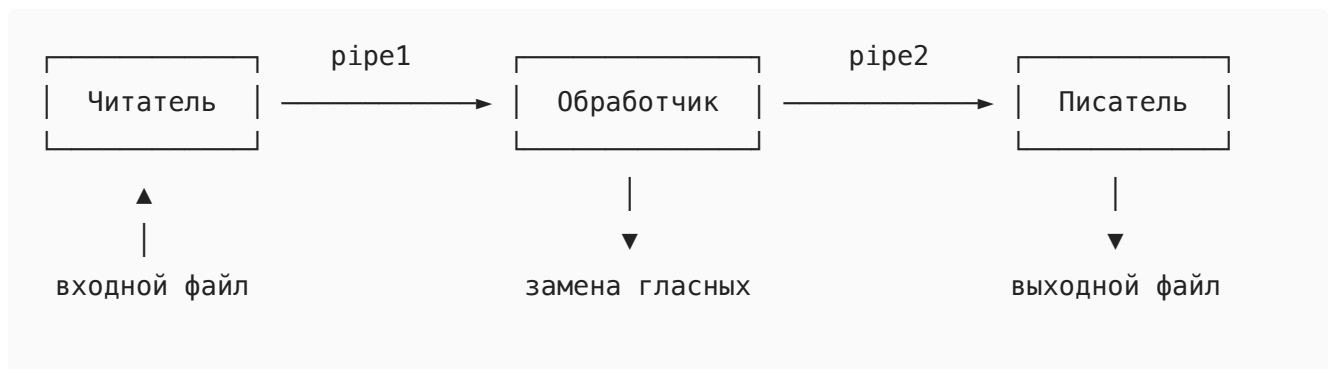
Названия программ записаны в формате `for_<grade>.c`, где `grade` - балл, на который выполнена программа.

Решение

Формат аргументов командной строки:

```
./<исполняемый файл> <входной файл> <выходной файл>
```

На 4 балла:



1. Первый процесс читает входной файл и передает данные во второй процесс через неименованный канал.
2. Второй процесс обрабатывает строку, заменяя все строчные гласные на заглавные, и передает данные в третий процесс.
3. Третий процесс записывает обработанные данные в выходной файл.
4. `pipe1` связывает Читателя и Обработчика.
5. `pipe2` связывает Обработчика и Писателя.

Имена входного и выходного файла передаются в аргументах командной строки.

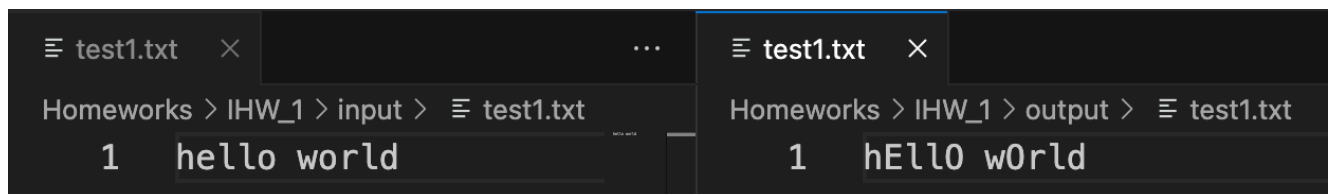
Ввод и вывод данных при работе с файлами осуществляется через системные вызовы операционной системы `read` и `write`.

Размер буфера - 5000 байт.

Программа работает с тестовыми данными, размер которых не превосходит размер буфера.

Тесты:

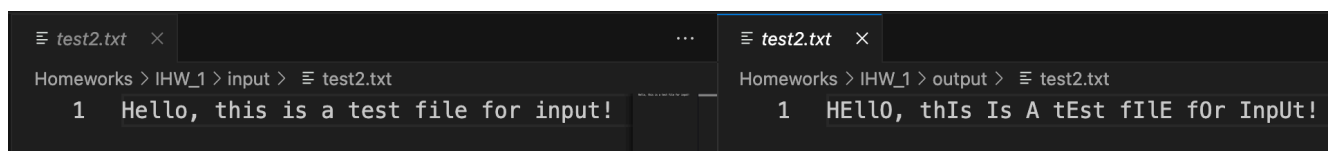
Тест 1



```
test1.txt x
Homeworks > IHW_1 > input > test1.txt
1 hello world

test1.txt x
Homeworks > IHW_1 > output > test1.txt
1 hEllO wOrld
```

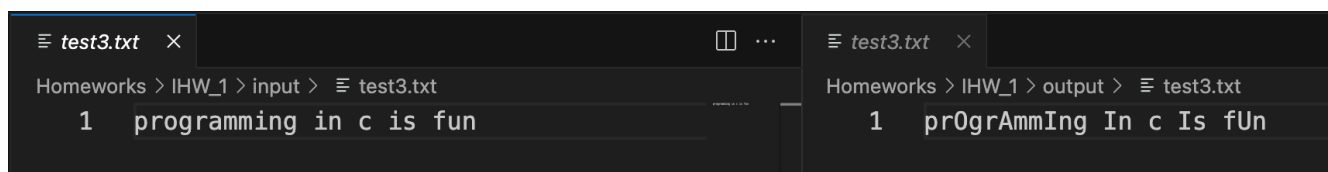
Тест 2



```
test2.txt x
Homeworks > IHW_1 > input > test2.txt
1 Hello, this is a test file for input!

test2.txt x
Homeworks > IHW_1 > output > test2.txt
1 HEllO, thIs Is A tEst fIlE fOr InPUt!
```

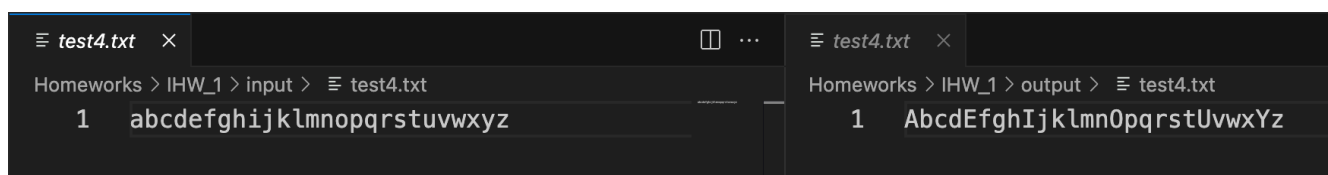
Тест 3



```
test3.txt x
Homeworks > IHW_1 > input > test3.txt
1 programming in c is fun

test3.txt x
Homeworks > IHW_1 > output > test3.txt
1 pr0grAmmIng In c Is fUn
```

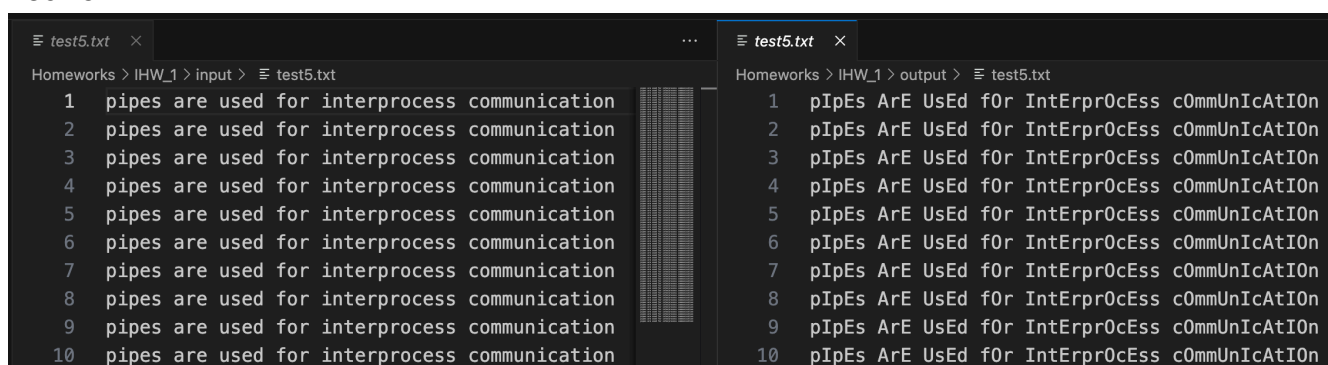
Тест 4



```
test4.txt x
Homeworks > IHW_1 > input > test4.txt
1 abcdefghijklmnopqrstuvwxyz

test4.txt x
Homeworks > IHW_1 > output > test4.txt
1 AbCdEfGhIjKlMnOpQrStUvWxYz
```

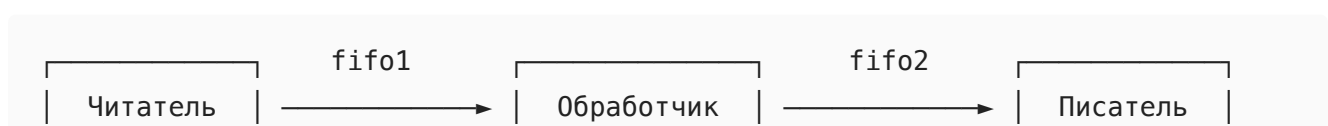
Тест 5



```
test5.txt x
Homeworks > IHW_1 > input > test5.txt
1 pipes are used for interprocess communication
2 pipes are used for interprocess communication
3 pipes are used for interprocess communication
4 pipes are used for interprocess communication
5 pipes are used for interprocess communication
6 pipes are used for interprocess communication
7 pipes are used for interprocess communication
8 pipes are used for interprocess communication
9 pipes are used for interprocess communication
10 pipes are used for interprocess communication

test5.txt x
Homeworks > IHW_1 > output > test5.txt
1 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
2 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
3 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
4 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
5 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
6 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
7 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
8 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
9 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
10 pIpEs ArE UsEd fOr IntErprOcEss c0mmUnIcAtI0n
```

На 5 баллов:





1. Читатель читает входной файл и передает данные через `fifo1`.
2. Обработчик получает данные из `fifo1`, заменяет гласные буквы и передает результат через `fifo2`.
3. Писатель получает данные из `fifo2` и записывает их в выходной файл.
4. Именованный канал `fifo1` связывает Читателя и Обработчик.
5. Именованный канал `fifo2` связывает Обработчик и Писателя.

Имена каналов и их создание:

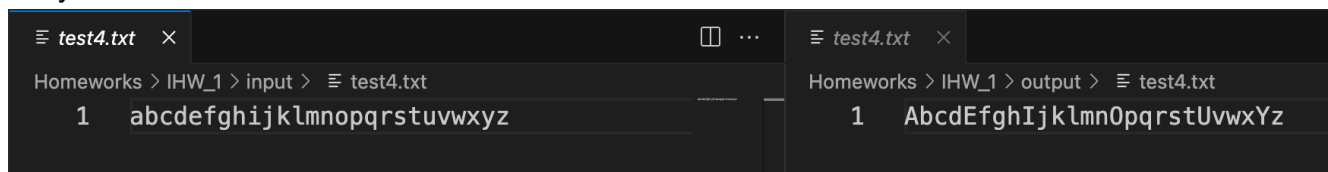
```
#define FIF01 "fifo1"
#define FIF02 "fifo2"

mkfifo(FIF01, 0666);
mkfifo(FIF02, 0666);
```

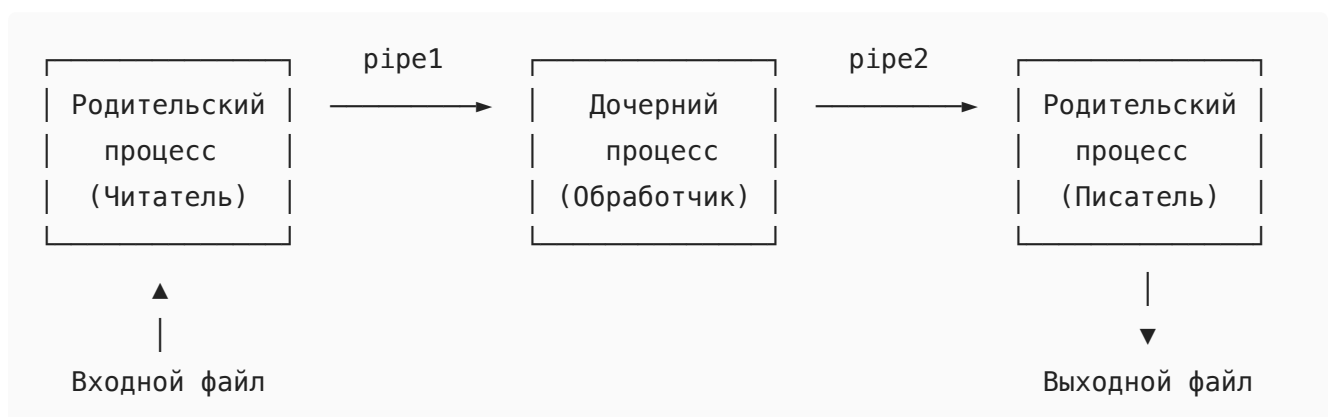
Всё остальное аналогично предыдущей программе.

Результат:

Результаты также аналогичны.



На 6 баллов:



1. Читатель - Родительский процесс:
 - Читает входной файл.
 - Передаёт данные дочернему процессу через `pipe1`.

2. Обработчик - Дочерний процесс:

- Получает данные из `pipe1`.
- Заменяет строчные гласные буквы на заглавные.
- Отправляет результат обратно родителю через `pipe2`.

3. Писатель - Родительский процесс:

- Получает обработанные данные через `pipe2`.
- Записывает результат в выходной файл.

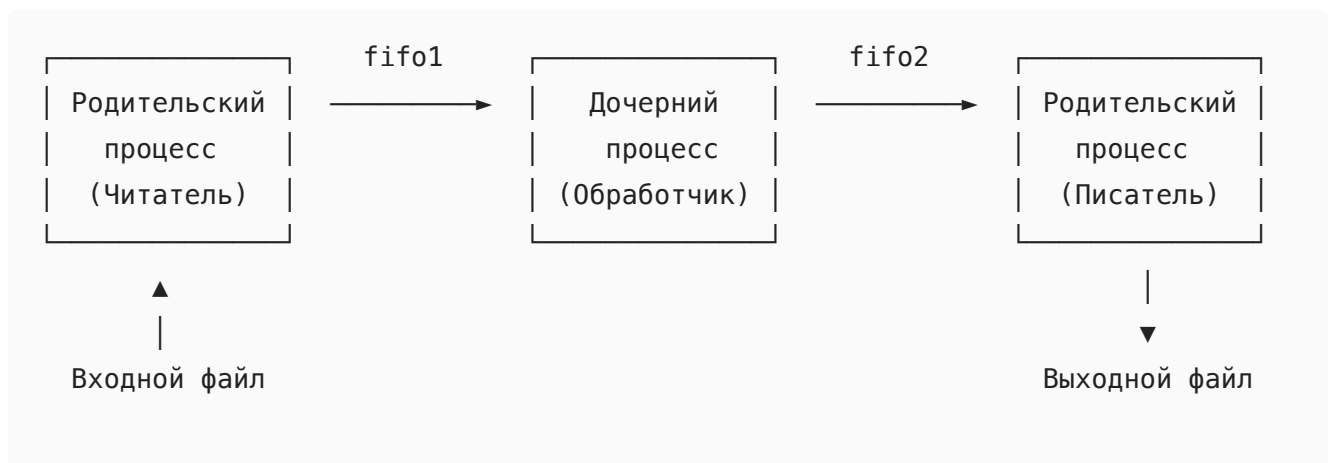
Результат:

```
for_6.c  test4.txt x
Homeworks > IHW_1 > input > test4.txt
1 abcdefghijklmnopqrstuvwxyz

test4.txt x
Homeworks > IHW_1 > output > test4.txt
1 AbcdEfghIjklmnOpqrstUvwYz
```

На остальных тестах аналогичные результаты.

На 7 баллов:



1. Создаются именованные каналы `FIFO1` и `FIFO2` с помощью `mkfifo()`.

2. Родительский процесс:

- Читает данные из входного файла.
- Записывает данные в `FIFO1`.
- Ждет обработки.
- Читает обработанные данные из `FIFO2`.
- Записывает результат в выходной файл.

3. Дочерний процесс:

- Читает данные из `FIFO1`.
- Заменяет строчные гласные буквы на заглавные.
- Записывает результат в `FIFO2`.

4. После завершения работы именованные каналы удаляются с помощью `unlink()`.

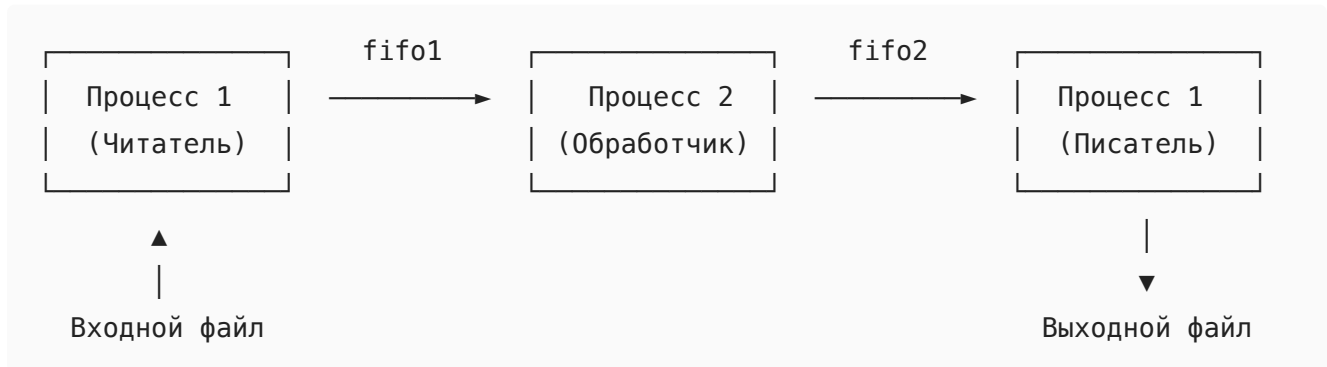
Результат:

```
for_7.c  test4.txt x
Homeworks > IHW_1 > input > test4.txt
1 abcdefghijklmnopqrstuvwxyz

test4.txt x
Homeworks > IHW_1 > output > test4.txt
1 AbcdEfghIjklmnOpqrstUvwXyz
```

Для остальных тестов результат аналогичен предыдущим программам.

На 8 баллов:



1. Первый процесс (process1):
 - Читает данные из входного файла.
 - Передает данные через именованный канал `fifo1` второму процессу.
 - Получает обработанные данные через `fifo2`.
 - Записывает результат в выходной файл.
2. Второй процесс (process2):
 - Получает данные из `fifo1`.
 - Заменяет строчные гласные буквы на заглавные.
 - Отправляет данные обратно через `fifo2`.

Пример запуска:

1. Компиляция

```
gcc for_8_process1.c -o process1
gcc for_8_process2.c -o process2
```

2. Запуск первого процесса с передачей файлов

```
./process1 input/test4.txt output/test4.txt
```

3. Запуск второго процесса с обработкой файла

```
./process2
```

Результат:

```
for_8_process1.c  for_8_process2.c  test4.txt  test4.txt
Homeworks > IHW_1 > input > test4.txt  Homeworks > IHW_1 > output > test4.txt
1  abcdefghijklmnopqrstuvwxyz  1  AbcdEfghIjklmnOpqrstUvwYz
```

Для остальных тестов результат аналогичен предыдущим программам.

На 9 баллов:

Данные программы отличаются от предыдущих только тем, что используется цикл `while` для чтения всех данных файла в буффер размера 128 байтов.

Результат:

```
test6.txt  test6.txt
Homeworks > IHW_1 > input > test6.txt  Homeworks > IHW_1 > output > test6.txt
1  abcdefghijklmnopqrstuvwxyzabcdefghijklmnopq  1  AbcdEfghIjklmnOpqrstUvwYzAbcdEfghIjklmnOpq
```

Этот тестовый входной файл содержит 10 повторяющихся элементов `abcdefghijklmnopqrstuvwxyz`, что равно 260 байтам.

На остальных тестах результаты также верные.

На 10 баллов:

Пример запуска:

1. Компиляция

```
gcc for_8_process1.c -o process1
gcc for_8_process2.c -o process2
```

1. Запуск первого процесса с входным и выходным файлом

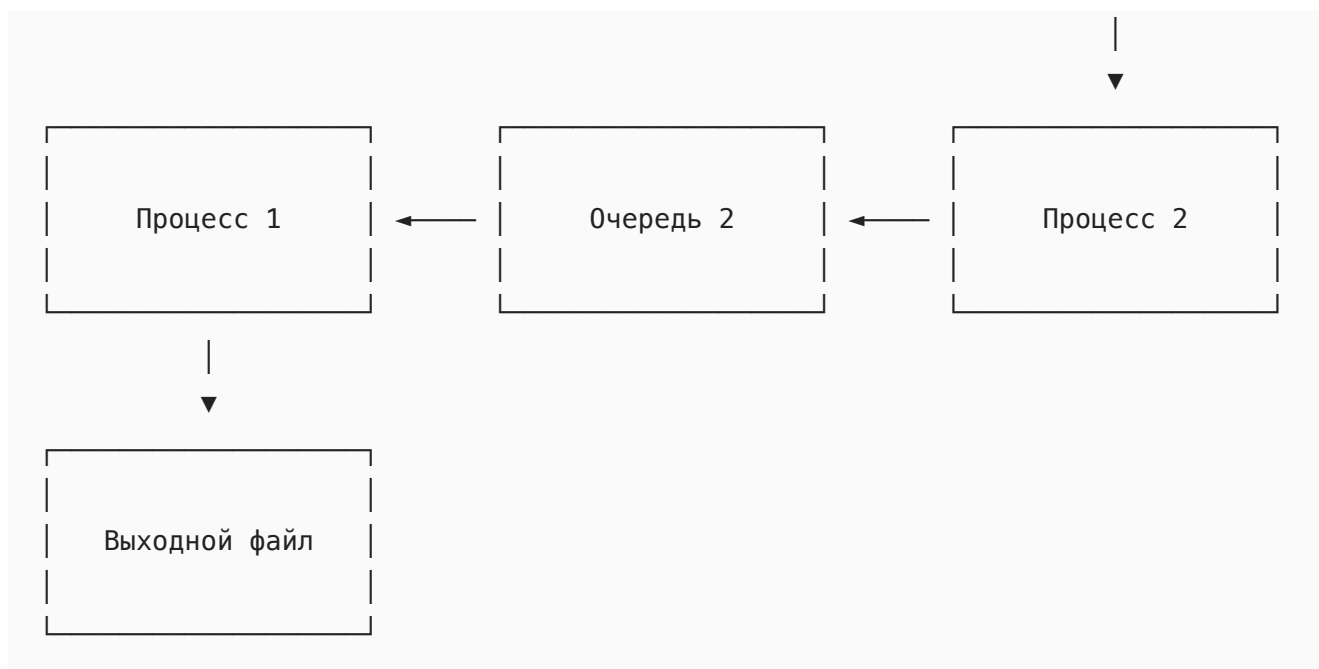
```
./process1 input/test6.txt output/test6.txt
```

3. Запуск второго процесса с обработкой файла

```
./process2
```

Схема решения задачи:





1. Процесс 1:

- Чтение данных из входного файла.
- Передача данных в 1 очередь сообщений.
- Получение обработанных данных из 2 очереди сообщений.
- Запись результата в выходной файл.

2. Процесс 2:

- Получение данных из 1 очереди сообщений.
- Замена строчных гласных буквы на заглавные.
- Передача обработанных данных во 2 очередь сообщений.

Результат:

```
for_10_process1.c  test6.txt x  test6.txt x
Homeworks > IHW_1 > input > test6.txt  Homeworks > IHW_1 > output > test6.txt
1 abcdefghijklmnopqrstuvwxyzabcdefghijklmnopqrstu  1 AbcdEfgHIjklmNopqrstUvwYzAbcdEfgHIjklmNopqrstU
```

Для остальных тестов результат аналогичен предыдущим программам.