Programming in C II, Group A
Jacobs University Bremen
Dr. Kinga Lipskoch

Course: JTSK-320112
February 23rd, 2018

Assignment 6 - Queues and Working with Files

- The problems of this assignment must be solved in C.
- Your programs should have the input and output formatting according to the testcases listed after the problems.
- Your programs should consider the grading rules:
  https://grader.eecs.jacobs-university.de/courses/320112/2018_1gA/Grading-Criteria-C2.pdf

Problem 6.1 *Counting a word in a file* (2 points)

Presence assignment, due by 18:30 h today

Write a program which reads the content of a file given as input and counts the occurrence of a given word in the file. The word counting should be case insensitive. It is assumed the words are separated by one or multiple of the following characters: ' ' ',' '?' '!' '.' '\t' '\r' '\n'. Do not count the cases when the word is a substring of another word.
For testing your solution to this problem, please use:
`http://jgrader.de/courses/320112/c/words.txt`
`http://jgrader.de/courses/320112/c/words2.txt`
You can assume that the content of the input file will be valid if existing.

Testcase 6.1: input

```
words.txt
is
```

Testcase 6.1: output

```
The file contains the word 'is' 3 times.
```

Problem 6.2 *Removing from the queue* (2 points)

Extend the source code of `queue.c` from **Problem 5.4** by implementing the `dequeue()` function. Follow the hints given in the slides (see Lecture 5 & 6, page 17) and consider the case of a queue underflow.
You can assume that the input will be valid except the semantical possibility of reaching queue underflow.

Testcase 6.2: input

```
1
a
1
c
1
f
2
2
4
```

Testcase 6.2: output

```
add char: Putting a into queue
1 items in queue
Type 1 to add, 2 to delete, 4 to quit:
add char: Putting c into queue
2 items in queue
Type 1 to add, 2 to delete, 4 to quit:
add char: Putting f into queue
3 items in queue
Type 1 to add, 2 to delete, 4 to quit:
Removing a from queue
2 items in queue
Type 1 to add, 2 to delete, 4 to quit:
Removing c from queue
1 items in queue
Type 1 to add, 2 to delete, 4 to quit:
Bye.
```

Problem 6.3 *Printing the queue* (2 points)

Extend the source code of `queue.h`, `queue.c` and `testqueue.c` from **Problem 6.2** by adding and implementing the additional function `printq()` for printing the elements of the queue separated by spaces. If you enter `3`, then the program should print the elements of the queue. Make sure that you cannot just print once, and that your program does not crash on the formulated condition.

You can assume that the input will be syntactically correct.

Testcase 6.3: input

```
1
a
1
c
1
f
3
4
```

Testcase 6.3: output

```
add char: Putting a into queue
1 items in queue
Type 1 to add, 2 to delete, 3 to print, 4 to quit:
add char: Putting c into queue
2 items in queue
Type 1 to add, 2 to delete, 3 to print, 4 to quit:
add char: Putting f into queue
3 items in queue
Type 1 to add, 2 to delete, 3 to print, 4 to quit:
content of the queue: a c f
3 items in queue
Type 1 to add, 2 to delete, 3 to print, 4 to quit:
Bye.
```

Bonus Problem 6.4 *Binary read and write* (2 points)

Write a program which reads using `fread` the first two characters (separated by space) from the file "`chars.txt`" and writes using `fwrite` the sum of their ASCII code values as an integer into "`codesum.txt`". Use an editor to create the input file "`chars.txt`". Your program is responsible to create the output file "`codesum.txt`".

You can safely assume that the content of the input file will be valid and the two characters are separated by a space.

Problem 6.5 *"Query a database" with files* (3 points)

Write a program which reads from the standard input a file name and a person's name, and then queries the age and the city corresponding to the previously read person's name.

The program should repeatedly return the age and the city corresponding to the name introduced from the standard input until the word "`exit`" is introduced. It is assumed that the word "`exit`" is not contained in the input file as a valid name. It is also assumed that a name occurs only once in the file.

Do not store the whole information (three arrays for the names, ages and cities) in the main memory, store only partial information (e.g., name) and use the functions `ftell()` and `fseek()` to read the age and city, which are always following the name in this order.

For testing your solution to this problem, please use:

`https://jgrader.de/courses/320112/c/persons.txt`

`https://jgrader.de/courses/320112/c/persons2.txt`

You can assume that the content of the input file will be valid if existing.

Testcase 6.5: input

```
persons.txt
ben
kate
bill
exit
```

Testcase 6.5: output

```
Age and city of ben:
21
hamburg
Age and city of kate:
20
hannover
Age and city of bill:
21
bremen
Exiting ...
```

Problem 6.6 *Copy n files* (3 points)

Write a program which reads from the standard input the value of an integer `n` and then the names of `n` files. The program should copy the content of the `n` files and write the result into `n` other files with the original name and the additional suffix `_copy`.
Read the input files and write the output files using the binary mode (`fread` and `fwrite`). Use a char buffer of size 64 bytes and chunks of size 1 byte when reading and the same buffer with chunks of size 64 bytes (or less if the last write and file size is not a multiply of 64) when writing.
For testing your solution to this problem, please use:
`https://jgrader.de/courses/320112/c/file1`
`https://jgrader.de/courses/320112/c/file2`
`https://jgrader.de/courses/320112/c/file3`
You can assume that the content of the input files will be valid if existing.

Testcase 6.6: input

```
3
file1
file2
file3
```

Testcase 6.6: output

```
Copied file1 into file1_copy.
Copied file2 into file2_copy.
Copied file3 into file3_copy.
```

How to submit your solutions

- Your source code should be properly indented and compile with `gcc` without any warnings (You can use `gcc -Wall -o program program.c`). Insert suitable comments (not on every line ...) to explain what your program does.
- Please name the programs according to the suggested filenames (they should match the description of the problem) in Grader. Otherwise you might have problems with the inclusion of header files.
  Each program **must** include a comment on the top like the following:

```
/*
   JTSK-320112
   a6_p1.c
   Firstname Lastname
   myemail@jacobs-university.de
*/
```

- You have to submit your solutions via *Grader* at
  `https://cantaloupe.eecs.jacobs-university.de`.
  If there are problems (but **only** then) you can submit the programs by sending mail to
  `k.lipskoch@jacobs-university.de` **with a subject line that begins with JTSK-320112**.
  It is important that you do begin your subject with the coursenumber, otherwise I might have problems to identify your submission.
- Please note, that after the deadline it will not be possible to submit any solutions. It is useless to send late solutions by mail, because they will not be accepted.

This assignment is due by Wednesday, February 28th, 10:00 h.