Programming in C II, Module II
Jacobs University Bremen
Dr. Jürgen Schönwälder

Course: JTSK-320112
February 16th, 2017

Assignment 3 - Linked Lists and Multiple Sources

- 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 posted on the course web page:
  `http://cnds.eecs.jacobs-university.de/courses/c2-2017/`

Problem 3.1 *A linked list* (1 point)

Presence assignment, due by 18:30 h today

Using the example from the slides (Lecture 3 & 4, pages $22 - 25$), write a program that uses a linked list. Your program should wait for input from the keyboard. Entering from the keyboard an 'a' will just add the following number (read as next from the keyboard) to the end of the list, while a 'b' inserts at the beginning of the list. The character 'r' will remove the first element from the list, a 'p' will print the list while a 'q' will free the memory used by the list and quit the execution of the program.
Use a switch-case statement to decide which action to take.
You can assume that the input will be valid.

Testcase 3.1: input

```
b
2
b
3
a
4
p
r
p
q
```

Testcase 3.1: output

```
3 2 4
2 4
```

Problem 3.2 *An enhanced linked list* (2 points)

Extend your program for **Problem 3.1** by writing a function for inserting a new element into the list at a given position and a function for reversing the order of the elements in the list. Your program should wait for input from the keyboard. An 'i' followed by two numbers (the position and the number to be inserted) should insert the second the number at position of the first number (the first element in the list has position 0). You can assume that the input does not contain any logical errors (e.g., 'i' is always followed by two numbers, and 'b' and 'a' are followed by one number). However, if the position for inserting is negative or is greater than the number of elements in the list then print on the standard output "Invalid position!". An 'R' should reverse the order of the elements in the list without allocating new nodes or using a doubly linked list (i.e., only with the use of pointers).
Use a switch-case statement to decide which action to take.

Testcase 3.2: input

```
b
2
b
3
a
4
p
r
p
i
1
5
p
i
4
11
R
p
q
```

Testcase 3.2: output

```
3 2 4
2 4
2 5 4
Invalid position!
4 5 2
```

Problem 3.3 *A doubly linked list of characters* (4 points)

Create a data type that implements a doubly linked list of characters. Write a program that tests your double linked list with the testcase given below. A from the keyboard entered integer value 1 will add the following character to the list to the beginning of the list, while a 2 will remove all elements with the given character from the list, a 3 will print the current list, while a 4 will print the elements of the list backwards. A 5 will empty the list, free the memory used by the doubly linked list and quit the execution of the program.
You can assume that the input does not contain logical errors (i.e., if the command is 2, you can assume that a character will follow). However, a character to be deleted may not be currently in the list. In this case, the message "The element is not in the list!" should be printed on the standard output.
Use a switch-case statement to decide which action to take.

Testcase 3.3: input

```
1
r
1
a
1
d
3
1
a
1
x
1
r
1
x
3
2
x
3
4
2
b
5
```

Testcase 3.3: output

```
d a r
x r x a d a r
r a d a r
r a d a r
The element is not in the list!
```

Problem 3.4 *Multiple sources* (1 point)

Modify your solution for **Problem 3.1** such that you separate your source code into three files: struct declaration and function declarations in `linked_list.h`, function definitions in `linked_list.c`, and your main function in `use_linked_list.c`.
You can assume that the input will be valid.

Testcase 3.4: input

```
b
2
b
3
a
4
p
r
p
q
```

Testcase 3.4: output

```
3 2 4
2 4
```

How to submit your solutions

- Your source code should be properly indented and compile with `gcc` without any warnings. You should use `gcc -Wall -Werror -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
   a3_p1.c
   Firstname Lastname
   myemail@jacobs-university.de
*/
```

- You have to submit your solutions via *Grader* at
  `https://grader.eecs.jacobs-university.de`.
  If there are problems (but **only** then) you can submit the programs by sending mail to `j.schoenwaelder@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 Tuesday, February 21st, 10:00 h.