

Advanced Programming COEN 11

Project 6

Project 6

- **Waiting List with File I/O**
 - Extension of project 4
 - Due in week 7

Project 6

- **Initially**
 - The waiting list may be either
 - empty
 - formed with information read from a file
- **At the end**
 - The updated waiting list is saved into a file

Project 6

- The info should be saved in a text file (in ascii) according to the following format:

Names	Number

Joe	5
Mary	3
Zoe	3

- It should be possible to read the file with commands such as cat and more

Project 6

- The name of the file is an argument for the program
 - If the file does not exist
 - fopen returns NULL for reading
 - the list starts empty and is saved at the end into a file with the given name
 - If the file does exist
 - the list is initially formed with the information obtained from the file and is saved into the same file at the end

Project 6

- The name of the file is an argument for the program
 - Example:
 - # ./wait_list <file_name>
 - or
 - # ./a.out <file_name>

Project 6

- The name of the file is the first argument for the program
 - In the code:

```
main (int argc, char *argv[ ])
{
    char *filename;
    if (argc == 1)
    {
        printf ("The name of the file is missing!\n");
        exit (1);
    }
    else
        filename = argv[1];
    ...
}
```

Project 6

- The name of the file is an argument for the program
 - In the code:
 - argc gives the number of arguments
 - argv is an array of strings, each of which is one of the arguments for the program
 - argv[0] is the name of the executable
 - argv[1] - argv[argc - 1] are the arguments

Project 6

- The waiting list is created interactively, as in project 4, except that command quit will save the info into a file.
 - insert name number - insert a node with the name and number of people specified
 - next number - extract (show and delete) oldest node with the corresponding number
 - list - print the list, name and number, from oldest to newest
 - quit - save the list in the file specified and quit

Project 6

- Use same insert function for inserting information from the file and from the keyboard.
- Your insert function should have the following interface:
void insert (char *name, int number);
- Read the name and number to local variables (char array and int) before calling the insert function.