Version: 1.0

Selection

C-File-Operations

Summary

Implement C programs to replicate common file operations, such as reading, writing, and manipulating files using system calls.

#C #FileHandling #SystemCalls

42

Intellectual Property Disclaimer

All content presented in this training module, including but not limited to texts, images, graphics, and other materials, is protected by intellectual property rights held by Association 42.

Terms of Use:

- **Personal use:** You are permitted to use the contents of this module solely for personal purpose. Any commercial use, reproduction, distribution, modification, or public display is strictly prohibited without prior written permission from Association 42.
- **Respect for Integrity:** You must not alter, transform, or adapt the content in any way that could harm its integrity.

Protection of Rights:

Any violation of these terms constitutes an infringement of intellectual property rights and may result in legal action. We reserve the right to take all necessary measures to protect our rights, including but not limited to claims for damages.

For any questions regarding the use of the content or to obtain authorization, please contact: `legal@42.fr`

Contents

1 Instructions 1

2 Foreword 5

3 Exercise 0: display_file 7

4 Exercise 1: cat 8

5 Exercise 2: tail 9

6 Exercise 3: hexdump 10

7 Submission and peer-evaluation 11

Chapter 1

Instructions

- Only this page will serve as a reference: do not trust rumors.
- Watch out! This document could potentially change up until submission.
- Make sure you have the appropriate permissions on your files and directories.
- You have to follow the submission procedures for all your exercises.
- Your exercises will be checked and graded by your fellow classmates.
- Additionally, your exercises will be checked and graded by a program called Moulinette.
- Moulinette is very meticulous and strict in its evaluation of your work. It is entirely automated, and there is no way to negotiate with it. So, to avoid bad surprises, be as thorough as possible.
- Moulinette is not very open-minded. It won't try to understand your code if it doesn't adhere to the Norm. Moulinette relies on a program called `norminette` to check if your files respect the norm. TL;DR: it would be foolish to submit work that doesn't pass `norminette`'s check.
- These exercises are carefully laid out by order of difficulty - from easiest to hardest. We `will not` consider a successfully completed harder exercise if an easier one is not perfectly functional.
- Using a forbidden function is considered cheating. Cheaters get -42, and this grade is non-negotiable.
- You'll only have to submit a main() function if we ask for a program.
- Moulinette compiles with these flags: -Wall -Wextra -Werror, and uses `cc`.
- If your program doesn't compile, you'll get `0`.
- You cannot leave any additional files in your directory other than those specified in the subject.
- Got a question? Ask your peer on the right. Otherwise, try your peer on the left.
- Your reference guide is called `Google / man / the Internet / ...`.
- Check out the Slack Piscine.

© 2026 Association 42. - All rights reserved

- Examine the examples thoroughly. They could very well call for details that are not explicitly mentioned in the subject...
- By Odin, by Thor! Use your brain!!!

Do not forget to add the *standard 42 header* in each of your .c/.h files. Norminette checks its existence anyway!

Norminette must be launched with the *-R CheckForbiddenSourceHeader* flag. Moulinette will use it too.

© 2026 Association 42. - All rights reserved

Context

The C Piscine is intense. It's your first big challenge at 42 — a deep dive into problem-solving, autonomy, and community.

During this phase, your main objective is to build your foundation — through struggle, repetition, and especially **peer-learning** exchange.

In the AI era, shortcuts are easy to find. However, it's important to consider whether your AI usage is truly helping you grow — or simply getting in the way of developing real skills.

The Piscine is also a human experience — and for now, nothing can replace that. Not even AI.

For a more complete overview of our stance on AI — as a learning tool, as part of the ICT curriculum, and as a growing expectation in the job market — please refer to the dedicated FAQ available on the intranet.

Main message

- 👉 Build strong foundations without shortcuts.
- 👉 Really develop tech & power skills.
- 👉 Experience real peer-learning, start learning how to learn and solve new problems.
- 👉 The learning journey is more important than the result.
- 👉 Learn about the risks associated with AI, and develop effective control practices and countermeasures to avoid common pitfalls.

Learner rules:

- You should apply reasoning to your assigned tasks, especially before turning to AI.
- You should not ask for direct answers to the AI.
- You should learn about 42 global approach on AI.

Phase outcomes:

Within this foundational phase, you will get the following outcomes:

- Get proper tech and coding foundations.
- Know why and how AI can be dangerous during this phase.

© 2026 Association 42. - All rights reserved

Comments and example:

- Yes, we know AI exists — and yes, it can solve your activities. But you're here to learn, not to prove that AI has learned. Don't waste your time (or ours) just to demonstrate that AI can solve the given problem.
- Learning at 42 isn't about knowing the answer — it's about developing the ability to find one. AI gives you the answer directly, but that prevents you from building your own reasoning. And reasoning takes time, effort, and involves failure. The path to success is not supposed to be easy.
- Keep in mind that during exams, AI is not available — no internet, no smartphones, etc. You'll quickly realise if you've relied too heavily on AI in your learning process.
- Peer learning exposes you to different ideas and approaches, improving your interpersonal skills and your ability to think divergently. That's far more valuable than just chatting with a bot. So don't be shy — talk, ask questions, and learn together!
- Yes, AI will be part of the curriculum — both as a learning tool and as a topic in itself. You'll even have the chance to build your own AI software. In order to learn more about our crescendo approach you'll go through in the documentation available on the intranet.

✓ Good practice:

> I'm stuck on a new concept. I ask someone nearby how they approached it. We talk for 10 minutes — and suddenly it clicks. I get it.

✗ Bad practice:

> I secretly use AI, copy some code that looks right. During peer evaluation, I can't explain anything. I fail. During the exam — no AI — I'm stuck again. I fail.

© 2026 Association 42. - All rights reserved

Chapter 2

Foreword

Here's a little story :

(1982, California) Larry Walters of Los Angeles is one of the few to contend for the Darwin Awards and live to tell the tale. "I have fulfilled my 20-year dream," said Walters, a former truck driver for a company that makes TV commercials. "I'm staying on the ground. I've proved the thing works."
Larry's boyhood dream was to fly. But fates conspired to keep him from his dream. He joined the Air Force, but his poor eyesight disqualified him from the job of pilot. After he was discharged from the military, he sat in his backyard watching jets fly overhead.

He hatched his weather balloon scheme while sitting outside in his "extremely comfortable" Sears lawnchair. He purchased 45 weather balloons from an Army-Navy surplus store, tied them to his tethered lawnchair (dubbed the Inspiration I) and filled the four-foot diameter balloons with helium. Then, armed with some sandwiches, Miller Lite, and a pellet gun, he strapped himself into his lawnchair. He figured he would shoot to pop a few of the many balloons when it was time to descend.

Larry planned to sever the anchor and lazily float to a height of about 30 feet above the backyard, where he would enjoy a few hours of flight before coming back down. But things didn't work out quite as Larry planned.

When his friends cut the cord anchoring the lawnchair to his Jeep, he did not float lazily up to 30 feet. Instead he streaked into the LA sky as if shot from a cannon, pulled by the lift of 45 helium balloons, holding 33 cubic feet of helium each.

He didn't level off at 100 feet, nor did he level off at 1000 feet. After climbing and climbing, he leveled off at 16,000 feet.

At that height he felt he couldn't risk shooting any of the balloons, lest he unbalance the load and really find himself in trouble. So he stayed there, drifting cold and frightened with his beer and sandwiches, for more than 14 hours. He crossed the primary approach corridor of LAX, where startled Trans World Airlines and Delta Airlines pilots radioed in reports of the strange sight.

© 2026 Association 42. - All rights reserved

Eventually he gathered the nerve to shoot a few balloons, and slowly descended. The hanging tethers tangled and caught in a power line, blacking out a Long Beach neighborhood for 20 minutes. Larry climbed to safety, where he was arrested by waiting members of the LAPD. As he was led away in handcuffs, a reporter dispatched to cover the daring rescue asked him why he had done it. Larry replied nonchalantly, "A man can’t just sit around."

The Federal Aviation Administration was not amused. Safety Inspector Neal Savoy said, "We know he broke some part of the Federal Aviation Act, and as soon as we decide which part it is, a charge will be filed."

The moral of this story is Larry Walters should have stay on his chair and learn C....

© 2026 Association 42. - All rights reserved

Chapter 3

Exercise 0: display_file

	Exercise: 0	
display_file		
Directory: *ex0/*		
Files to Submit: `Makefile, and files needed for your program`		
Authorized: `close, open, read, write`		

- Create a program called `ft_display_file` that displays, on the standard output, only the content of the file given as argument.
- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`. The binary will be called `ft_display_file`.
- The `malloc` function is forbidden. You can only do this exercise by declaring a fixed-sized array.
- All files given as arguments will be valid.
- Error messages have to be displayed on their reserved output followed by a new line.

If no argument is given, it should display:

```
File name missing.
```

If there is more than one argument, it should display:

```
Too many arguments.
```

If the file cannot be read, it should display:

```
Cannot read file.
```

© 2026 Association 42. - All rights reserved

Chapter 4

Exercise 1: cat

	Exercise: 1	
cat		
Directory: *ex1/*		
Files to Submit: `Makefile, and files needed for your program`		
Authorized: `close, open, read, write, strerror, basename`		

- Create a program called `ft_cat` which does the same thing as the system's `cat` command-line.
- You don't have to handle options.
- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`.
- You may use the variable `errno` (check the `man` for `Errno`).
- You should read the man of all the authorized functions
- You can only do this exercise by declaring a fixed-sized array. This array will have a size limited to a little less than 30 `ko`. In order to test that size-limit, use the `ulimit` command-line in your Shell.

© 2026 Association 42. - All rights reserved

Chapter 5

Exercise 2: tail

	Exercise: 2	
tail		
Directory: *ex2/*		
Files to Submit: `Makefile, and files needed for your program`		
Authorized: `close, open, read, write, malloc, free, strerror, basename`		

- Create a program called `ft_tail` which does the same thing as the system command `tail`.
- The only option you have to handle is `-c`, but you don't need to handle '+' or '-' signs.
- all the test will be done with the `-c` option.
- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`.
- You may use the variable `errno`.

© 2026 Association 42. - All rights reserved

Chapter 6

Exercise 3: hexdump

	Exercise: 3	
hexdump		
Directory: *ex3/*		
Files to Submit: `Makefile, and files needed for your program`		
Authorized: `close, open, read, write, malloc, free, strerror, basename`		

- Create a program called `ft_hexdump` which does the same thing as the system's `hexdump` command-line without redirection.
- The only option you have to handle is `-C`.
- The submission directory should have a `Makefile` with the following rules : `all`, `clean`, `fclean`.
- You may use the variable `errno`.

© 2026 Association 42. - All rights reserved

Chapter 7

Submission and peer-evaluation

Turn in your assignment in your `Git` repository as usual. Only the work inside your repository will be evaluated during the defense. Don't hesitate to double check the names of your files to ensure they are correct.

You need to return only the files requested by the subject of this project.

© 2026 Association 42. - All rights reserved