Version: 1.0

Selection

C-Strings

Summary

Implement fundamental C string functions, including validation, transformation, and concatenation operations.

#C #Strings #Manipulation

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

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.

- 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.

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.

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.

Chapter 2

Foreword

Morty: Rick!

Rick: Uhp-uhp-uhp! Morty, keep your hands off your ding-dong! It's the only way we can speak freely. Look around you, Morty. Do you really think this wuh-world is real? You'd have to be an idiot not to notice all the sloppy details. Look, that guy's putting a bun between two hot dogs.

Morty: I dunno, Rick, I mean, I've seen people do that before.

Rick: Well, look at that old lady. She's-she's walking a cat on a leash.

Morty: Uh, Mrs. Spencer does that all the time, Rick.

Rick: Look, I-I-I don't want to hear about Mrs. Spencer, Morty! She's an idiot! All right, all right, there. Wh-what about that, Morty?

Morty: Okay, okay, you got me on that one.

Rick: Oh, really, Morty? Are you sure you haven't seen that somewhere in real life before?

Morty: No, no, I haven't seen that. I mean, why would a Pop-Tart want to live inside a toaster, Rick? I mean, th-that would be like the scariest place for them to live. Y'know what I mean?

Rick: You're missing the point, Morty. Why would he drive a smaller toaster with wheels? I mean, does your car look like a smaller version of your house? No.

Morty: So, why are they doing this? W-what do they want?

Rick: Well, that would be obvious to you, Morty, if you'd been paying attention.
[an ambulance drives past Rick and Morty and stops; open back doors]
Paramedic: We got the President of the United States in here! We need 10cc of concentrated dark matter, stat, or he'll die!

Morty: Concentrated dark matter? They were asking about that in class.

Rick: Yeah, it’s a special fuel I invented to travel through space faster than anybody else. These Zigerions are always trying to scam me out of my secrets, but they made a big mistake this time, Morty. They dragged you into this. Now they’re gonna pay!

Morty: What do you- w-w-what are we gonna do?

Rick: We’re gonna scam the scammers, Morty. And we’re gonna take ’em for everything they’ve got.

The following exercices will be easier to complete if you are a fan of "Rick and Morty"

Chapter 3

Exercise 0: ft_strcpy

	Exercise: 0	
ft_strcpy		
Directory: *ex0/*		
Files to Submit: `ft_strcpy.c`		
Authorized: `None`		

- Reproduce the behavior of the function `strcpy` (man strcpy).

Prototype:

```
char *ft_strcpy(char *dest, char *src);
```

Chapter 4

Exercise 1: ft_strncpy

	Exercise: 1	
ft_strncpy		
Directory: *ex1/*		
Files to Submit: `ft_strncpy.c`		
Authorized: `None`		

- Reproduce the behavior of the function `strncpy` (man strncpy).

Prototype:

```
char *ft_strncpy(char *dest, char *src, unsigned int n);
```

Chapter 5

Exercise 2: ft_strcmp

	Exercise: 2	
ft_strcmp		
Directory: *ex2/*		
Files to Submit: `ft_strcmp.c`		
Authorized: `None`		

- Reproduce the behavior of the function `strcmp` (man strcmp).

Prototype:

```
int ft_strcmp(char *s1, char *s2);
```

Chapter 6

Exercise 3: ft_strncmp

	Exercise: 3	
ft_strncmp		
Directory: *ex3/*		
Files to Submit: `ft_strncmp.c`		
Authorized: `None`		

- Reproduce the behavior of the function `strncmp` (man strncmp).

Prototype:

```
int ft_strncmp(char *s1, char *s2, unsigned int n);
```

Chapter 7

Exercise 4: ft_strcat

	Exercise: 4	
ft_strcat		
Directory: *ex4/*		
Files to Submit: `ft_strcat.c`		
Authorized: `None`		

- Reproduce the behavior of the function `strcat` (man strcat).

Prototype:

```
char *ft_strcat(char *dest, char *src);
```

Chapter 8

Exercise 5: ft_strncat

	Exercise: 5	
ft_strncat		
Directory: *ex*5/		
Files to Submit: `ft_strncat.c`		
Authorized: `None`		

- Reproduce the behavior of the function `strncat` (man strncat).

Prototype:

```
char *ft_strncat(char *dest, char *src, unsigned int nb);
```

Chapter 9

Exercise 6: ft_strstr

	Exercise: 6	
ft_strstr		
Directory: *ex6/*		
Files to Submit: `ft_strstr.c`		
Authorized: `None`		

- Reproduce the behavior of the function `strstr` (man strstr).

Prototype:

```
char *ft_strstr(char *str, char *to_find);
```

Chapter 10

Exercise 7: ft_atoi

	Exercise: 7	
ft_atoi		
Directory: *ex7/*		
Files to Submit: `ft_atoi.c`		
Authorized: `None`		

- Write a function that converts the initial portion of the string pointed by `str` to its integer representation.
- The string can start with an arbitrary amount of white space (as determined by `isspace(3)`).
- The string can be preceded by an arbitrary amount of + and - signs.
- A - sign will change the sign of the integer returned depending on whether the number of - is odd or even.
- The function should read the string until it encounters a non-digit character and return the number found so far.
- You do not need to handle overflow or underflow; the result can be undefined in such cases.
- Here's an example of a program that prints the return value of `ft_atoi`:

Example:

```
$>./a.out " --+-+1234ab567"
-1234
```

Prototype:

```
int ft_atoi(char *str);
```

Chapter 11

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.