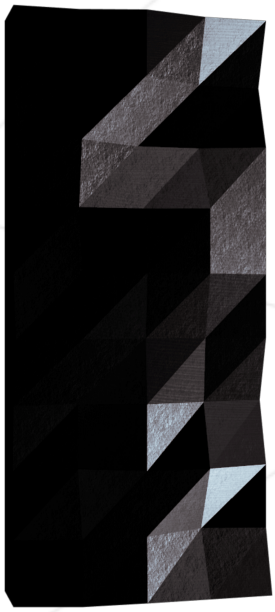

C Bootcamp

Mini-project 02 : evalexpr

Staff WeThinkCode_ info@wethinkcode.co.za

Summary: Third mini-project of the C Bootcamp @ WeThinkCode_.

Contents

Chapter I

Foreword

Here is what `Wikipedia` has to say about Pinkie Pie :

```
Pinkie Pie (fully named Pinkamena Diane Pie) is a pink earth pony based on
the "G3" toy of the same name. Her character, summarized by Thiessen as
"a frenetic sugar rush", was inspired by the "G1" pegasus toy Surprise.
She works as a live-in party planner at Sugarcube Corner, a bakery and
confectionery store in Ponyville that resembles a gingerbread house,
where she keeps a toothless baby alligator named Gummy. A comedic character
raised on a "dreary rock farm", Pinkie is cheerful, energetic, and talkative.
She is defined by her desire to entertain her friends by throwing parties
at random times and acting as outlandish as possible; however, she
demonstrates a lack of confidence and a fear of being rejected by others,
which is occasionally expressed by her balloon-like mane deflating. Pinkie
is a source of much of the series' humor,[20] and several of the show's
"wacky gags" are kept exclusive to her. Her running gags include
breaking the fourth wall and "appearing suddenly in unexpected places",
as well as an ability to predict future events through various body reactions,
which she calls the "Pinkie Sense". In early episodes, Faust worked to depict
Pinkie as a "free spirit" to address concerns of the character being seen as
too "hyper" and "ditzy". As the creative team grew more comfortable with
Pinkie's character and humor, she became "really over-the-top strange
and bordering on crazy, with a wacky cartoonish magic all her own.
```

Don't forger `sharing kindness, it's an easy feat.`

Chapter II

Instructions

- Only this page will serve as reference: do not trust rumors.
- Watch out! This document could potentially change up to an hour before 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.
- On top of that, 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 if you want to avoid bad surprises, be as thorough as you can be.
- Moulinette is not very open-minded. It won't try and understand your code if it doesn't respect the Norm. Moulinette relies on a program called `Norminator` to check if your files respect the norm. TL;DR: it would be idiotic to submit a piece of work that doesn't pass `Norminator`'s check.
- Using a forbidden function is considered cheating. Cheaters get `-42`, and this grade is non-negotiable.
- If ft_putchar() is an authorized function, we will compile your code with our `ft_putchar.c`.
- You'll only have to submit a main() function if we ask for a program.
- These exercises are carefully laid out by order of difficulty - from easiest to hardest. We `will not` take into account a successfully completed harder exercise if an easier one is not perfectly functional.

- Moulinette compiles with these flags: -Wall -Wextra -Werror, and uses `gcc`.
- If your program doesn't compile, you'll get `0`.
- Exercises in shell have to be executed with /bin/sh.
- You cannot leave any additional file in your directory than those specified in the subject.
- Got a question? Ask your peer on your right. Otherwise, try your peer on your left.
- Your reference guide is called `Google / man / the Internet / ...`.
- Check out the "C Bootcamp" part of the forum on the intranet.
- 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 !!!

Chapter III

Subject

	Exercise 00
eval_expr	
Turn-in directory : $ex00/$	
Files to turn in : `Makefile, and all of your program's files`	
Allowed functions : `write, malloc, free`	
Notes : `n/a`	

- Create a program called `eval_expr`.
- It'll have a function `eval_expr` prototyped as follows :

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

- This function takes a characters string as argument. This string represents an arithmetic expression. For example :

  `"3 + 42 * (1 - 2 / (3 + 4) - 1 % 21) + 1"`

- This expression must be calculated and its result must be returned.
- The string passed as argument will be valid (no bugs, no bogus addresses, no letters or syntax errors, no division by zero, etc...).
- The following five operators must be supported :
  - `+` for addition
  - `-` for subtraction
  - `/` for division

- `*` for multiplication
- `%` for modulo

- The function must also support any amount of brackets.
- Here's your `main` :

```
int main(int ac, char **av)
{
  if (ac > 1)
  {
    ft_putnbr(eval_expr(av[1]));
    ft_putchar('\n');
  }
  return (0);
}
```