Assignment name : ft\_scanf

Expected files : ft\_scanf.c

Allowed functions: fgetc, ungetc, ferror, feof, isspace, isdigit, stdin, va\_start, va\_arg, va\_copy, va\_end

--------------------------------------------------------------------------------

Write a function named `ft\_scanf` that will mimic the real scanf with the following constraints:

- It will manage only the following conversions: s, d, and c

- You don't have to handle the options \*, m and '

- You don't have to handle the maximum field width

- You don't have to handle the types modifier characters (h, hh, l, etc.)

- You don't have to handle the conversions beginning with %n$

Your function must be declared as follows:

int ft\_scanf(const char \*, ... );

You will find in this directory a file containing a part of the code you will need, you just have to complete it.

To test your program compare your results with the real scanf.

Hint : You may need to read the man of scanf.

#include <stdarg.h>

#include <stdio.h>

#include <ctype.h>

int match\_space(FILE \*f)

{

// You may insert code here

return (0);

}

int match\_char(FILE \*f, char c)

{

// You may insert code here

return (0);

}

int scan\_char(FILE \*f, va\_list ap)

{

// You may insert code here

return (0);

}

int scan\_int(FILE \*f, va\_list ap)

{

// You may insert code here

return (0);

}

int scan\_string(FILE \*f, va\_list ap)

{

// You may insert code here

return (0);

}

int match\_conv(FILE \*f, const char \*\*format, va\_list ap)

{

switch (\*\*format)

{

case 'c':

return scan\_char(f, ap);

case 'd':

match\_space(f);

return scan\_int(f, ap);

case 's':

match\_space(f);

return scan\_string(f, ap);

case EOF:

return -1;

default:

return -1;

}

}

int ft\_vfscanf(FILE \*f, const char \*format, va\_list ap)

{

int nconv = 0;

int c = fgetc(f);

if (c == EOF)

return EOF;

ungetc(c, f);

while (\*format)

{

if (\*format == '%')

{

format++;

if (match\_conv(f, &format, ap) != 1)

break;

else

nconv++;

}

else if (isspace(\*format))

{

if (match\_space(f) == -1)

break;

}

else if (match\_char(f, \*format) != 1)

break;

format++;

}

if (ferror(f))

return EOF;

return nconv;

}

int ft\_scanf(const char \*format, ...)

{

// ...

int ret = ft\_vfscanf(stdin, format, ap);

// ...

return ret;

}