0x04 more functions nested loops

README

**C - More functions, more nested loops**

[**0. isupper**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/0-isupper.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023408118%26amp;usg%3DAOvVaw3X9evrpdF9DoowE48t0Gc5&sa=D&source=docs&ust=1703185023468937&usg=AOvVaw19tW-C9-md6HsMrBbBLKRk)

Write a function that checks for uppercase character.

* Prototype: int isupper(int c);
* Returns 1 if c is uppercase
* Returns 0 otherwise

[**1. isdigit**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/1-isdigit.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023408746%26amp;usg%3DAOvVaw0YG0HkPid9AKVseo3aoSB5&sa=D&source=docs&ust=1703185023469297&usg=AOvVaw3kDH_enRp8boKMeiAqxBm2)

Write a function that checks for a digit (0 through 9).

* Prototype: int isdigit(int c);
* Returns 1 if c is a digit
* Returns 0 otherwise

[**2. Collaboration is multiplication**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/2-mul.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023409477%26amp;usg%3DAOvVaw1WKeENfZWPQozTepw-A8yY&sa=D&source=docs&ust=1703185023469562&usg=AOvVaw2EKIUT3QMRpwjwke5UqcrD)

Write a function that multiplies two integers.

* Prototype: int mul(int a, int b);

[**3. The numbers speak for themselves**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/3-print_numbers.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023409987%26amp;usg%3DAOvVaw2S8gdPGNHw8F9HCwGEG-Qw&sa=D&source=docs&ust=1703185023469748&usg=AOvVaw2K5sPL6sMgxrvL_MrYubHo)

Write a function that prints the numbers, from 0 to 9, followed by a new line.

* Prototype: void print\_numbers(void);
* You can only use \_putchar twice in your code

[**4. I believe in numbers and signs**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/4-printmostnumbers.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023410720%26amp;usg%3DAOvVaw3b7ij_13ITOLSwyj6NQbBg&sa=D&source=docs&ust=1703185023470017&usg=AOvVaw3EQrvhL5p1N4StukKxZTE6)

Write a function that prints the numbers, from 0 to 9, followed by a new line.

* Prototype: void print\_most\_numbers(void);
* Do not print 2 and 4
* You can only use \_putchar twice in your code

[**5. Numbers constitute the only universal language**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/5-morenumbers.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023411339%26amp;usg%3DAOvVaw3g-PP9aDAEXVziK4SRIBg-&sa=D&source=docs&ust=1703185023470432&usg=AOvVaw2keuD_t4O_a7k6WEg1C_t7)

Write a function that prints 10 times the numbers, from 0 to 14, followed by a new line.

* Prototype: void more\_numbers(void);
* You can only use \_putchar three times in your code

[**6. The shortest distance between two points is a straight line**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/6-printline.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023411876%26amp;usg%3DAOvVaw26d9VA-z54XyWyx6HTIFyN&sa=D&source=docs&ust=1703185023470685&usg=AOvVaw0n6bgcQTZV4YI4ocIiJaiD)

Write a function that draws a straight line in the terminal.

* Prototype: void print\_line(int n);
* You can only use \_putchar function to print
* Where n is the number of times the character \_ should be printed
* The line should end with a \n
* If n is 0 or less, the function should only print \n

[**7. I feel like I am diagonally parked in a parallel universe**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/7-printdiagonal.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023412744%26amp;usg%3DAOvVaw2EHeaGi-zQFOIaUdq_352K&sa=D&source=docs&ust=1703185023470972&usg=AOvVaw36M3XYzlrRi2Lw_jd-eXZp)

Write a function that draws a diagonal line on the terminal.

* Prototype: void print\_diagonal(int n);
* You can only use \_putchar function to print
* Where n is the number of times the character \ should be printed
* The diagonal should end with a \n
* If n is 0 or less, the function should only print a \n

[**8. You are so much sunshine in every square inch**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/8-printsquare.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023413809%26amp;usg%3DAOvVaw19M7wiK7am4MYX4wPM3LYv&sa=D&source=docs&ust=1703185023471436&usg=AOvVaw1GAox4XhYu6hyRO32PHb8h)

Write a function that prints a square, followed by a new line.

* Prototype: void print\_square(int size);
* You can only use \_putchar function to print
* Where size is the size of the square
* If size is 0 or less, the function should print only a new line
* Use the character # to print the square

[**9. Fizz-Buzz**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/9-fizzbuzz.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023414699%26amp;usg%3DAOvVaw2odonIMruuRVTqq9iMvBdE&sa=D&source=docs&ust=1703185023471955&usg=AOvVaw2Lzhg6tq129OTqIOrDSYb5)

The “Fizz-Buzz test” is an interview question designed to help filter out the 99.5% of programming job candidates who can’t seem to program their way out of a wet paper bag.

\*Write a program that prints the numbers from 1 to 100, followed by a new line. But for multiples of three print Fizz instead of the number and for the multiples of five print Buzz. For numbers which are multiples of both three and five print FizzBuzz.

* Each number or word should be separated by a space
* You are allowed to use the standard library

[**10. Triangles**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/10-printtriangle.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023415435%26amp;usg%3DAOvVaw2RhHwzFQg8IyG3r60x5NOD&sa=D&source=docs&ust=1703185023472266&usg=AOvVaw05LnYKVdIXEngPqBIiW0u_)

Write a function that prints a triangle, followed by a new line.

* Prototype: void print\_triangle(int size);
* You can only use \_putchar function to print
* Where size is the size of the triangle
* If size is 0 or less, the function should print only a new line
* Use the character # to print the triangle

[**11. The problem of distinguishing prime numbers from composite numbers and of resolving the latter into their prime factors is known to be one of the most important and useful in arithmetic**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/100-primefactor.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023416379%26amp;usg%3DAOvVaw0flk0z_93F2txUWKmRVp0P&sa=D&source=docs&ust=1703185023472618&usg=AOvVaw0Zi1s2UYhtYKeXpQ0gi-th)

The prime factors of 1231952 are 2, 2, 2, 2, 37 and 2081.

Write a program that finds and prints the largest prime factor of the number 612852475143, followed by a new line.

* You are allowed to use the standard library
* Your program will be compiled with this command: gcc -Wall -pedantic -Werror -Wextra -std=gnu89 100-prime\_factor.c -o 100-prime\_factor -lm

[**12. Numbers have life; they're not just symbols on paper**](https://www.google.com/url?q=https://www.google.com/url?q%3Dhttps://github.com/KipronohVincent/alx-low_level_programming/blob/master/0x04-more_functions_nested_loops/101-print_number.c%26amp;sa%3DD%26amp;source%3Deditors%26amp;ust%3D1703185023417112%26amp;usg%3DAOvVaw2Mdpx5NWYtZos-d7u2fe4G&sa=D&source=docs&ust=1703185023472863&usg=AOvVaw1_GFX34EOIiV8wsJ7s5K_u)

Write a function that prints an integer.

* Prototype: void print\_number(int n);
* You can only use \_putchar function to print
* You are not allowed to use long
* You are not allowed to use arrays or pointers
* You are not allowed to hard-code special values

===================================================

MAIN.H CODE

#ifndef MAIN\_H

#define MAIN\_H

int \_putchar(char c);

int \_isupper(int c);

int \_isdigit(int c);

int mul(int a, int b);

void print\_numbers(void);

void print\_most\_numbers(void);

void more\_numbers(void);

void print\_line(int n);

void print\_diagonal(int n);

void print\_square(int size);

void print\_triangle(int size);

void print\_number(int n);

#endif

===================================================

\_PUTCHAR CODE

#include "main.h"

#include <unistd.h>

/\*\*

 \* \_putchar - writes the character c to stdout

 \* @c: The character to print

 \*

 \* Return: On success 1.

 \* On error, -1 is returned, and errno is set appropriately.

 \*/

int \_putchar(char c)

{

 return (write(1, &c, 1));

}

===================================================

0-isupper.c CODE

#include "main.h"

/\*\*

\* \_isupper - checks for uppercase character

\* @c: Variable text

\* Return: Always 0.

\*/

int \_isupper(int c)

{

 if (c >= 'A' && c <= 'Z')

 {

 return (1);

 }

 else

{

 return (0);

}

}

==================================================

1-isdigit.c CODE

#include "main.h"

/\*\*

 \* \_isdigit - checks for a digit (0 through 9)

 \* @c: int to be checked

 \* Return: 1 if c is a digit, 0 otherwise

 \*/

int \_isdigit(int c)

{

 return (c >= '0' && c <= '9');

}

===========================================

2-mul.c CODE

#include "main.h"

/\*\*

 \* mul - checks for checks for a digit (0 through 9).

 \* @a: a - Variable

 \* @b: b - variable

 \* Return: Always 0.

 \*/

int mul(int a, int b)

{

 int mul;

 mul = a \* b;

 return (mul);

}

================================================

3-print\_numbers.c CODE

#include "main.h"

/\*\*

 \* print\_numbers - checks for checks for a digit (0 through 9).

 \*

 \* Return: Always 0.

 \*/

void print\_numbers(void)

{

 int c;

 for (c = 48; c < 58; c++)

 {

 \_putchar(c);

 }

 \_putchar('\n');

}

==============================================

4-print\_most\_numbers.c CODE

#include "main.h"

/\*\*

 \* print\_most\_numbers - checks for checks for a digit (0 through 9).

 \*

 \* Return: Always 0.

 \*/

void print\_most\_numbers(void)

{

 int c;

 for (c = 48; c < 58; c++)

 {

 if (c != 50)

 {

 if (c != 52)

 {

 \_putchar(c);

 }

 }

 }

 \_putchar('\n');

}

============================================

5-more\_numbers.c CODE

#include "main.h"

/\*\*

 \* more\_numbers - prints 10 times the numbers, from 0 to 14

 \* followed by a new line

 \*/

void more\_numbers(void)

{

 int i, j;

 for (i = 0; i < 10; i++)

 {

 for (j = 0; j < 15; j++)

 {

 if (j >= 10)

 \_putchar(j / 10 + '0');

 \_putchar(j % 10 + '0');

 }

 \_putchar('\n');

 }

}

================================================

6-print\_line.c CODE

#include "main.h"

/\*\*

 \* print\_line - draws a straight line in the terminal

 \* @n: number of times the character \_ should be printed

 \*/

void print\_line(int n)

{

 if (n <= 0)

 {

 \_putchar('\n');

 } else

 {

 int i;

 for (i = 1; i <= n; i++)

 {

 \_putchar('\_');

 }

 \_putchar('\n');

 }

}

==============================================

7-print\_diagonal.c CODE

#include "main.h"

/\*\*

 \* print\_diagonal - draws a diagonal line on the terminal

 \* @n: number of times the character \ should be printed

 \*/

void print\_diagonal(int n)

{

 if (n <= 0)

 {

 \_putchar('\n');

 } else

 {

 int i, j;

 for (i = 0; i < n; i++)

 {

 for (j = 0; j < n; j++)

 {

 if (j == i)

 \_putchar('\\');

 else if (j < i)

 \_putchar(' ');

 }

 \_putchar('\n');

 }

 }

}

=============================================

8-print\_square.c CODE

#include "main.h"

/\*\*

 \* print\_square - prints a square, followed by a new line;

 \* @size: size of the square

 \*/

void print\_square(int size)

{

 if (size <= 0)

 {

 \_putchar('\n');

 } else

 {

 int i, j;

 for (i = 0; i < size; i++)

 {

 for (j = 0; j < size; j++)

 {

 \_putchar('#');

 }

 \_putchar('\n');

 }

 }

}

===============================================

9-fizz\_buzz.c CODE

#include "main.h"

#include <stdio.h>

/\*\*

 \* main - prints the numbers from 1 to 100, followed by a new line

 \* but for multiples of three prints Fizz instead of the number

 \* and for the multiples of five prints Buzz

 \* Return: Always 0 (Success)

 \*/

int main(void)

{

 int i;

 for (i = 1; i <= 100; i++)

 {

 if (i % 3 == 0 && i % 5 != 0)

 {

 printf(" Fizz");

 } else if (i % 5 == 0 && i % 3 != 0)

 {

 printf(" Buzz");

 } else if (i % 3 == 0 && i % 5 == 0)

 {

 printf(" FizzBuzz");

 } else if (i == 1)

 {

 printf("%d", i);

 } else

 {

 printf(" %d", i);

 }

 }

 printf("\n");

 return (0);

}

==============================================

10-print\_triangle.c CODE

#include "main.h"

/\*\*

 \* print\_triangle - prints a triangle, followed by a new line

 \* @size: size of the triangle

 \*/

void print\_triangle(int size)

{

 if (size <= 0)

 {

 \_putchar('\n');

 } else

 {

 int i, j;

 for (i = 1; i <= size; i++)

 {

 for (j = i; j < size; j++)

 {

 \_putchar(' ');

 }

 for (j = 1; j <= i; j++)

 {

 \_putchar('#');

 }

 \_putchar('\n');

 }

 }

}

===========================================

100-prime\_factor.c CODE

#include <stdio.h>

#include <math.h>

/\*\*

 \* main - finds and prints the largest prime factor of the number 612852475143

 \* followed by a new line

 \* Return: Always 0 (Success)

 \*/

int main(void)

{

 long int n;

 long int max;

 long int i;

 n = 612852475143;

 max = -1;

 while (n % 2 == 0)

 {

 max = 2;

 n /= 2;

 }

 for (i = 3; i <= sqrt(n); i = i + 2)

 {

 while (n % i == 0)

 {

 max = i;

 n = n / i;

 }

 }

 if (n > 2)

 max = n;

 printf("%ld\n", max);

 return (0);

}

=============================================

101-print\_number.c CODE

#include "main.h"

/\*\*

 \* print\_number - prints an integer

 \* @n: integer to be printed

 \*/

void print\_number(int n)

{

 unsigned int n1;

 if (n < 0)

 {

 n1 = -n;

 \_putchar('-');

 } else

 {

 n1 = n;

 }

 if (n1 / 10)

 {

 print\_number(n1 / 10);

 }

 \_putchar((n1 % 10) + '0');

}