(/)

Curriculum
SE Foundations
Average: **108.76%**

0x01. Python - if/else, loops, functions

Python

By: Guillaume

Weight: 1

Project over - took place from May 9, 2023 6:00 AM to May 10, 2023 6:00 AM

An auto review will be launched at the deadline

In a nutshell…

- **Auto QA review:** 135.0/160 mandatory & 0.0/41 optional
- **Altogether: 84.38%**
  - Mandatory: 84.38%
  - Optional: 0.0%
  - Calculation: 84.38% + (84.38% * 0.0%) == **84.38%**

```
#!/usr/bin/python3

for i in range(1, 10):
    if i <= 5:
        print("AAAAAHHHHH")
    elif i >= 6 and i <= 8:
        print("WHERE ARE ALL THE BRACKETS???")
    else:
        print("HOW DO YOU PEOPLE READ THIS SYNTAX EASILY")
```

Help

Resources

Read or watch:

- More Control Flow Tools (/rltoken/jpjs5EnZTpBLLEremJYjPQ) (*Read until "4.6. Defining Functions" included*)
- IndentationError (/rltoken/F9n2AE-fpEPzt2PfBMGYAQ)
- How To Use String Formatters in Python 3 (/rltoken/ZdtRIAkFu8dMBT99DcFBNg)
- Learn to Program (/rltoken/ElQgZYNHrLI7kV_ysEB1hQ)
- Learn to Program 2 : Looping (/rltoken/ElQgZYNHrLI7kV_ysEB1hQ)
- Pycodestyle – Style Guide for Python Code (/rltoken/TuTTnEg_Rwn8U1g3PEsZmA)

man or help:

- `python3`

Learning Objectives

At the end of this project, you are expected to be able to explain to anyone (/rltoken/SdBJUMTPS5VW3cQNkhaeSg), **without the help of Google**:

General

- Why Python programming is awesome
- Why indentation is so important in Python
- How to use the `if`, `if ... else` statements
- How to use comments
- How to affect values to variables
- How to use the `while` and `for` loops
- How is Python's `for` different from `C` 's?
- How to use the `break` and `continues` statements
- How to use `else` clauses on loops
- What does the `pass` statement do, and when to use it
- How to use `range`
- What is a function and how do you use functions
- What does return a function that does not use any `return` statement
- Scope of variables
- What's a traceback
- What are the arithmetic operators and how to use them

Copyright - Plagiarism

- You are tasked to come up with solutions for the tasks below yourself to meet with the above learning objectives.
- You will not be able to meet the objectives of this or any following project by copying and pasting someone else's work.
- You are not allowed to publish any content of this project.
- Any form of plagiarism is strictly forbidden and will result in removal from the program.

Requirements

Python Scripts

- Allowed editors: `vi, vim, emacs`
- All your files will be interpreted/compiled on Ubuntu 20.04 LTS using python3 (version 3.8.5)
- All your files should end with a new line
- The first line of all your files should be exactly `#!/usr/bin/python3`
- A `README.md` file, at the root of the folder of the project, is mandatory
- Your code should use the pycodestyle (version `2.8.*`)
- All your files must be executable
- The length of your files will be tested using `wc`

C Scripts

- Allowed editors: `vi, vim, emacs`
- All your files will be compiled on Ubuntu 20.04 LTS using gcc, using the options -Wall -Werror -Wextra -pedantic -std=gnu89
- All your files should end with a new line
- Your code should use the `Betty` style. It will be checked using betty-style.pl (https://github.com/alx-tools/Betty/blob/master/betty-style.pl) and betty-doc.pl (https://github.com/alx-tools/Betty/blob/master/betty-doc.pl)
- You are not allowed to use global variables
- No more than 5 functions per file
- In the following examples, the `main.c` files are shown as examples. You can use them to test your functions, but you don't have to push them to your repo (if you do we won't take them into account). We will use our own `main.c` files at compilation. Our `main.c` files might be different from the one shown in the examples
- The prototypes of all your functions should be included in your header file called `lists.h`
- Don't forget to push your header file
- All your header files should be include guarded

More Info

Note: you do not need to understand lists yet.

Quiz questions

Great! You've completed the quiz successfully! Keep going! (Show quiz)

Tasks

0. Positive anything is better than negative nothing

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

This program will assign a random signed number to the variable `number` each time it is executed. Complete the source code in order to print whether the number stored in the variable `number` is positive or negative.

- You can find the source code here (/rltoken/e4tR3cjFHqhelf4y485-zQ)
- The variable `number` will store a different value every time you will run this program
- You don't have to understand what `import`, `random. randint` do. Please do not touch this code
- The output of the program should be:
  - The number, followed by
    - if the number is greater than 0: `is positive`
    - if the number is 0: `is zero`
    - if the number is less than 0: `is negative`
  - followed by a new line

```
guillaume@ubuntu:~/0x01$ ./0-positive_or_negative.py
-4 is negative
guillaume@ubuntu:~/0x01$ ./0-positive_or_negative.py
0 is zero
guillaume@ubuntu:~/0x01$ ./0-positive_or_negative.py
-3 is negative
guillaume@ubuntu:~/0x01$ ./0-positive_or_negative.py
-10 is negative
guillaume@ubuntu:~/0x01$ ./0-positive_or_negative.py
10 is positive
guillaume@ubuntu:~/0x01$ ./0-positive_or_negative.py
-5 is negative
guillaume@ubuntu:~/0x01$ ./0-positive_or_negative.py
6 is positive
guillaume@ubuntu:~/0x01$ ./0-positive_or_negative.py
7 is positive
guillaume@ubuntu:~/0x01$ ./0-positive_or_negative.py
5 is positive
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `0-positive_or_negative.py`

(/)Done! | Help | Check your code | >_ Get a sandbox | QA Review

1. The last digit

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

This program will assign a random signed number to the variable `number` each time it is executed. Complete the source code in order to print the last digit of the number stored in the variable `number`.

- You can find the source code here (/rltoken/Vku0ZPFeDPuXUKD8nZ4mOQ)
- The variable `number` will store a different value every time you will run this program
- You don't have to understand what `import`, `random.randint` do. **Please do not touch this code**. This line should not change: `number = random.randint(-10000, 10000)`
- The output of the program should be:
  - The string `Last digit of`, followed by
  - the number, followed by
  - the string `is`, followed by the last digit of `number`, followed by
    - if the last digit is greater than 5: the string `and is greater than 5`
    - if the last digit is 0: the string `and is 0`
    - if the last digit is less than 6 and not 0: the string `and is less than 6 and not 0`
  - followed by a new line

```
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of 4205 is 5 and is less than 6 and not 0
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of -626 is -6 and is less than 6 and not 0
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of 1144 is 4 and is less than 6 and not 0
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of -9200 is 0 and is 0
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of 5247 is 7 and is greater than 5
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of -9318 is -8 and is less than 6 and not 0
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of 3369 is 9 and is greater than 5
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of -5224 is -4 and is less than 6 and not 0
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of -4485 is -5 and is less than 6 and not 0
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of 3850 is 0 and is 0
guillaume@ubuntu:~/0x01$ ./1-last_digit.py
Last digit of 5169 is 9 and is greater than 5
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `1-last_digit.py`

Done! | Help | Check your code | Get a sandbox | QA Review

2. I sometimes suffer from insomnia. And when I can't fall asleep, I play what I call the alphabet game

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a program that prints the ASCII alphabet, in lowercase, not followed by a new line.

- You can only use one `print` function with string format
- You can only use one loop in your code
- You are not allowed to store characters in a variable
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x01$ ./2-print_alphabet.py
abcdefghijklmnopqrstuvwxyzguillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `2-print_alphabet.py`

Done! | Help | Check your code | Get a sandbox | QA Review

3. When I was having that alphabet soup, I never thought that it would pay off

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a program that prints the ASCII alphabet, in lowercase, not followed by a new line.

- Print all the letters except `q` and `e`
- You can only use one `print` function with string format
- You can only use one loop in your code
- You are not allowed to store characters in a variable
- You are not allowed to import any module

(7)
```
guillaume@ubuntu:~/0x01$ ./3-print_alphabt.py
abcdfghijklmnoprstuvwxyzguillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `3-print_alphabt.py`

Done! | Help | Check your code | Get a sandbox | QA Review

4. Hexadecimal printing

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a program that prints all numbers from `0` to `98` in decimal and in hexadecimal (as in the following example)

- You can only use one `print` function with string format
- You can only use one loop in your code
- You are not allowed to store numbers or strings in a variable
- You are not allowed to import any module

(7)

```
guillaume@ubuntu:~/0x01$ ./4-print_hexa.py
0 = 0x0
1 = 0x1
2 = 0x2
3 = 0x3
4 = 0x4
5 = 0x5
6 = 0x6
7 = 0x7
8 = 0x8
9 = 0x9
10 = 0xa
11 = 0xb
12 = 0xc
13 = 0xd
14 = 0xe
15 = 0xf
16 = 0x10
17 = 0x11
18 = 0x12
...
96 = 0x60
97 = 0x61
98 = 0x62
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `4-print_hexa.py`

☑ Done! | Help | Check your code | >_ Get a sandbox | QA Review

5. 00...99

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a program that prints numbers from `0` to `99`.

- Numbers must be separated by `,`, followed by a space
- Numbers should be printed in ascending order, with two digits
- The last number should be followed by a new line
- You can only use no more than 2 `print` functions with string format
- You can only use one loop in your code
- You are not allowed to store numbers or strings in a variable
- You are not allowed to import any module

(7)

```
guillaume@ubuntu:~/0x01$ ./5-print_comb2.py
00, 01, 02, 03, 04, 05, 06, 07, 08, 09, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20,
21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41,
42, 43, 44, 45, 46, 47, 48, 49, 50, 51, 52, 53, 54, 55, 56, 57, 58, 59, 60, 61, 62,
63, 64, 65, 66, 67, 68, 69, 70, 71, 72, 73, 74, 75, 76, 77, 78, 79, 80, 81, 82, 83,
84, 85, 86, 87, 88, 89, 90, 91, 92, 93, 94, 95, 96, 97, 98, 99
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `5-print_comb2.py`

☑ Done! | Help | Check your code | >_ Get a sandbox | QA Review

6. Inventing is a combination of brains and materials. The more brains you use, the less material you need

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a program that prints all possible different combinations of two digits.

- Numbers must be separated by `, `, followed by a space
- The two digits must be different
- `01` and `10` are considered the same combination of the two digits `0` and `1`
- Print only the smallest combination of two digits
- Numbers should be printed in ascending order, with two digits
- The last number should be followed by a new line
- You can only use no more than 3 `print` functions with string format
- You can only use no more than 2 loops in your code
- You are not allowed to store numbers or strings in a variable
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x01$ ./6-print_comb3.py
01, 02, 03, 04, 05, 06, 07, 08, 09, 12, 13, 14, 15, 16, 17, 18, 19, 23, 24, 25, 26,
27, 28, 29, 34, 35, 36, 37, 38, 39, 45, 46, 47, 48, 49, 56, 57, 58, 59, 67, 68, 69,
78, 79, 89
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`

- File: `6-print_comb3.py`

(/)

Done? | Help | Check your code | Ask for a new correction | Get a sandbox | QA Review

7. islower

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a function that checks for lowercase character.

- Prototype: `def islower(c):`
- Returns `True` if `c` is lowercase
- Returns `False` otherwise
- You are not allowed to import any module
- You are not allowed to use `str.upper()` and `str.isupper()`
- Tips: ord() (/rltoken/WglAv9ep-gg2wwo49DYfKg)

You don't need to understand `__import__`

```
guillaume@ubuntu:~/0x01$ cat 7-main.py
#!/usr/bin/env python3
islower = __import__('7-islower').islower

print("a is {}".format("lower" if islower("a") else "upper"))
print("H is {}".format("lower" if islower("H") else "upper"))
print("A is {}".format("lower" if islower("A") else "upper"))
print("3 is {}".format("lower" if islower("3") else "upper"))
print("g is {}".format("lower" if islower("g") else "upper"))

guillaume@ubuntu:~/0x01$ ./7-main.py
a is lower
H is upper
A is upper
3 is upper
g is lower
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `7-islower.py`

Done! | Help | Check your code | Get a sandbox | QA Review

8. To uppercase

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a function that prints a string in uppercase followed by a new line.

- Prototype: `def uppercase(str):`
- You can only use no more than 2 `print` functions with string format
- You can only use one loop in your code
- You are not allowed to import any module
- You are not allowed to use `str.upper()` and `str.isupper()`
- Tips: ord() (/rltoken/WglAv9ep-gg2wwo49DYfKg)

You don't need to understand `__import__`

```
guillaume@ubuntu:~/0x01$ cat 8-main.py
#!/usr/bin/env python3
uppercase = __import__('8-uppercase').uppercase

uppercase("best")
uppercase("Best School 98 Battery street")

guillaume@ubuntu:~/0x01$ ./8-main.py
BEST
BEST SCHOOL 98 BATTERY STREET
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `8-uppercase.py`

Done! | Help | Check your code | Get a sandbox | QA Review

9. There are only 3 colors, 10 digits, and 7 notes; it's what we do with them that's important

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a function that prints the last digit of a number.

- Prototype: `def print_last_digit(number):`

- Returns the value of the last digit
- You are not allowed to import any module

You don't need to understand `__import__`

```
guillaume@ubuntu:~/0x01$ cat 9-main.py
#!/usr/bin/env python3
print_last_digit = __import__('9-print_last_digit').print_last_digit

print_last_digit(98)
print_last_digit(0)
r = print_last_digit(-1024)
print(r)

guillaume@ubuntu:~/0x01$ ./9-main.py
8044
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `9-print_last_digit.py`

Done! | Help | Check your code | Get a sandbox | QA Review

10. a + b

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a function that adds two integers and returns the result.

- Prototype: `def add(a, b):`
- Returns the value of `a + b`
- You are not allowed to import any module

You don't need to understand `__import__`

(7)

```
guillaume@ubuntu:~/0x01$ cat 10-main.py
#!/usr/bin/env python3
add = __import__('10-add').add

print(add(1, 2))
print(add(98, 0))
print(add(100, -2))

guillaume@ubuntu:~/0x01$ ./10-main.py
3
98
98
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `10-add.py`

Done! | Help | Check your code | Get a sandbox | QA Review

11. a ^ b

mandatory

Score: 100.0% (*Checks completed: 100.0%*)

Write a function that computes `a` to the power of `b` and return the value.

- Prototype: `def pow(a, b):`
- Returns the value of `a ^ b`
- You are not allowed to import any module

You don't need to understand `__import__`

(7)

```
guillaume@ubuntu:~/0x01$ cat 11-main.py
#!/usr/bin/env python3
pow = __import__('11-pow').pow

print(pow(2, 2))
print(pow(98, 2))
print(pow(98, 0))
print(pow(100, -2))
print(pow(-4, 5))

guillaume@ubuntu:~/0x01$ ./11-main.py
4
9604
1
0.0001
-1024
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `11-pow.py`

☑ Done! | Help | Check your code | >_ Get a sandbox | QA Review

12. Fizz Buzz

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that prints the numbers from 1 to 100 separated by a space.

- For multiples of three print `Fizz` instead of the number and for multiples of five print `Buzz`.
- For numbers which are multiples of both three and five print `FizzBuzz`.
- Prototype: `def fizzbuzz():`
- Each element should be followed by a space
- You are not allowed to import any module

You don't need to understand `__import__`

(7)

```
guillaume@ubuntu:~/0x01$ cat 12-main.py
#!/usr/bin/env python3
fizzbuzz = __import__('12-fizzbuzz').fizzbuzz

fizzbuzz()
print("")

guillaume@ubuntu:~/0x01$ ./12-main.py | cat -e
1 2 Fizz 4 Buzz Fizz 7 8 Fizz Buzz 11 Fizz 13 14 FizzBuzz 16 17 Fizz 19 Buzz Fizz 22
23 Fizz Buzz 26 Fizz 28 29 FizzBuzz 31 32 Fizz 34 Buzz Fizz 37 38 Fizz Buzz 41 Fizz
43 44 FizzBuzz 46 47 Fizz 49 Buzz Fizz 52 53 Fizz Buzz 56 Fizz 58 59 FizzBuzz 61 62
Fizz 64 Buzz Fizz 67 68 Fizz Buzz 71 Fizz 73 74 FizzBuzz 76 77 Fizz 79 Buzz Fizz 82
83 Fizz Buzz 86 Fizz 88 89 FizzBuzz 91 92 Fizz 94 Buzz Fizz 97 98 Fizz Buzz $
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `12-fizzbuzz.py`

Done? | Help | Check your code | Ask for a new correction | Get a sandbox | QA Review

13. Insert in sorted linked list

mandatory

Score: 0.0% (*Checks completed: 0.0%*)

Technical interview preparation:

- You are not allowed to google anything
- Whiteboard first

Write a function in C that inserts a number into a sorted singly linked list.

- Prototype: `listint_t *insert_node(listint_t **head, int number);`
- Return: the address of the new node, or `NULL` if it failed

(7)

```
carrie@ubuntu:0x01$ cat lists.h
#ifndef LISTS_H
#define LISTS_H

/**
 * struct listint_s - singly linked list
 * @n: integer
 * @next: points to the next node
 *
 * Description: singly linked list node structure
 *
 */
typedef struct listint_s
{
    int n;
    struct listint_s *next;
} listint_t;

size_t print_listint(const listint_t *h);
listint_t *add_nodeint_end(listint_t **head, const int n);
void free_listint(listint_t *head);

listint_t *insert_node(listint_t **head, int number);

#endif /* LISTS_H */
```

(/)

```
carrie@ubuntu:0x01$ cat linked_lists.c
#include <stdio.h>
#include <stdlib.h>
#include "lists.h"

/**
 * print_listint - prints all elements of a listint_t list
 * @h: pointer to head of list
 * Return: number of nodes
 */
size_t print_listint(const listint_t *h)
{
    const listint_t *current;
    unsigned int n; /* number of nodes */

    current = h;
    n = 0;
    while (current != NULL)
    {
        printf("%i\n", current->n);
        current = current->next;
        n++;
    }

    return (n);
}

/**
 * add_nodeint_end - adds a new node at the end of a listint_t list
 * @head: pointer to pointer of first node of listint_t list
 * @n: integer to be included in new node
 * Return: address of the new element or NULL if it fails
 */
listint_t *add_nodeint_end(listint_t **head, const int n)
{
    listint_t *new;
    listint_t *current;

    current = *head;

    new = malloc(sizeof(listint_t));
    if (new == NULL)
        return (NULL);

    new->n = n;
    new->next = NULL;

    if (*head == NULL)
        *head = new;
    else
    {
        while (current->next != NULL)
```

```
            current = current->next;
        current->next = new;
    }

    return (new);
}

/**
 * free_listint - frees a listint_t list
 * @head: pointer to list to be freed
 * Return: void
 */
void free_listint(listint_t *head)
{
    listint_t *current;

    while (head != NULL)
    {
        current = head;
        head = head->next;
        free(current);
    }
}
```

(/)

```
carrie@ubuntu:0x01$ cat 13-main.c
#include <stdlib.h>
#include <string.h>
#include <stdio.h>
#include "lists.h"

/**
 * main - check the code for
 *
 * Return: Always 0.
 */
int main(void)
{
    listint_t *head;

    head = NULL;
    add_nodeint_end(&head, 0);
    add_nodeint_end(&head, 1);
    add_nodeint_end(&head, 2);
    add_nodeint_end(&head, 3);
    add_nodeint_end(&head, 4);
    add_nodeint_end(&head, 98);
    add_nodeint_end(&head, 402);
    add_nodeint_end(&head, 1024);
    print_listint(head);

    printf("-----------------\n");

    insert_node(&head, 27);

    print_listint(head);

    free_listint(head);

    return (0);
}
```

```
carrie@ubuntu:0x01$ gcc -Wall -Werror -Wextra -pedantic -std=gnu89 13-main.c linked_
lists.c 13-insert_number.c -o insert
carrie@ubuntu:0x01$ ./insert
0
1
2
3
4
98
402
1024
-----------------
0
1
2
3
4
27
98
402
1024
carrie@ubuntu:0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `13-insert_number.c, lists.h`

Done? | Help | Check your code | Ask for a new correction | Get a sandbox | QA Review

14. Smile in the mirror

#advanced

Score: 0.0% (*Checks completed: 0.0%*)

Write a program that prints the ASCII alphabet, in reverse order, alternating lowercase and uppercase (`z` in lowercase and `Y` in uppercase) , not followed by a new line.

- You can only use one `print` function with string format
- You can only use one loop in your code
- You are not allowed to store characters in a variable
- You are not allowed to import any module

```
guillaume@ubuntu:~/0x01$ ./100-print_tebahpla.py
zYxWvUtSrQpOnMlKjIhGfEdCbAguillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `100-print_tebahpla.py`

Done? | Help | Check your code | Ask for a new correction | Get a sandbox | QA Review

15. Remove at position

#advanced

Score: 0.0% (*Checks completed: 0.0%*)

Write a function that creates a copy of the string, removing the character at the position `n` (not the Python way, the "C array index").

- Prototype: `def remove_char_at(str, n):`
- You are not allowed to import any module

You don't need to understand `__import__`

```
guillaume@ubuntu:~/0x01$ cat 101-main.py
#!/usr/bin/env python3
remove_char_at = __import__('101-remove_char_at').remove_char_at

print(remove_char_at("Best School", 3))
print(remove_char_at("Chicago", 2))
print(remove_char_at("C is fun!", 0))
print(remove_char_at("School", 10))
print(remove_char_at("Python", -2))

guillaume@ubuntu:~/0x01$ ./101-main.py
Bes School
Chcago
 is fun!
School
Python
guillaume@ubuntu:~/0x01$
```

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `101-remove_char_at.py`

Done? | Help | Check your code | Ask for a new correction | Get a sandbox | QA Review

16. ByteCode -> Python #2 (7)

#advanced

Score: 0.0% (*Checks completed: 0.0%*)

Write the Python function `def magic_calculation(a, b, c):` that does exactly the same as the following Python bytecode:

```
  3           0 LOAD_FAST                0 (a)
              3 LOAD_FAST                1 (b)
              6 COMPARE_OP               0 (<)
              9 POP_JUMP_IF_FALSE       16

  4          12 LOAD_FAST                2 (c)
             15 RETURN_VALUE

  5     >>   16 LOAD_FAST                2 (c)
             19 LOAD_FAST                1 (b)
             22 COMPARE_OP               4 (>)
             25 POP_JUMP_IF_FALSE       36

  6          28 LOAD_FAST                0 (a)
             31 LOAD_FAST                1 (b)
             34 BINARY_ADD
             35 RETURN_VALUE

  7     >>   36 LOAD_FAST                0 (a)
             39 LOAD_FAST                1 (b)
             42 BINARY_MULTIPLY
             43 LOAD_FAST                2 (c)
             46 BINARY_SUBTRACT
             47 RETURN_VALUE
```

tips - ByteCode (/rltoken/BO9a7nq6424lGmtmwyB4cQ)

Repo:

- GitHub repository: `alx-higher_level_programming`
- Directory: `0x01-python-if_else_loops_functions`
- File: `102-magic_calculation.py`

Done? | Help | Check your code | Ask for a new correction | Get a sandbox | QA Review

(/)