**Source code for 0x01-python-if\_else\_loops\_functions**

**README.md**

0x01-python-if\_else\_loops\_functions

**Lists.h**

#ifndef LISTS\_H

#define LISTS\_H

#include <stdlib.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 \*/

**0-Positive-or-negative.py**

#!/usr/bin/python3

import random

number = random.randint(-10, 10)

if number > 0:

    print(f"{number} is positive")

elif number == 0:

    print(f"{number} is zero")

else:

    print(f"{number} is negative")

**1-last\_digit.py**

#!/usr/bin/python3

import random

number = random.randint(-10000, 10000)

if number < 0:

    lastdigit = number % -10

else:

    lastdigit = number % 10

if lastdigit > 5:

    print("Last digit of {:d} is {:d} and is greater than 5"

          .format(number, lastdigit))

elif lastdigit < 6 and lastdigit != 0:

    print("Last digit of {:d} is {:d} and is less than 6 and not 0"

          .format(number, lastdigit))

else:

    print("Last digit of {:d} is 0 and is 0".format(number))

**2-print\_alphabet.py**

#!/usr/bin/python3

for i in range(ord('a'), ord('z') + 1):

    print('{:c}'.format(i), end='')

**3-print\_alphabet.py**

#!/usr/bin/python3

for i in range(ord('a'), ord('z') + 1):

    if chr(i) != 'e' and chr(i) != 'q':

        print('{:c}'.format(i), end='')

**4-print\_hexa.py**

#!/usr/bin/python3

for num in range(0, 99):

    print('{} = 0x{:x}'.format(num, num))

**5-print\_comb2.py**

#!/usr/bin/python3

for num in range(0, 99):

    print('{:02d}, '.format(num), end='')

print('99')

**6-print\_comb3.py**

#!/usr/bin/python3

for x in range(0, 10):

    for y in range(x + 1, 10):

        if x == 8 and y == 9:

            print('89')

        else:

            print('{}{}, '.format(x, y), end='')

**7-islower.py**

#!/usr/bin/python3

def islower(c):

    if ord(c) >= 97 and ord(c) <= 122:

        return True

    else:

        return False

**8-uppercase.py**

#!/usr/bin/python3

def uppercase(str):

    for i in str:

        if ord(i) >= 97 and ord(i) <= 122:

            i = chr(ord(i) - 32)

        print("{}".format(i), end="")

    print()

**9-print\_last\_digit.py**

#!/usr/bin/python3

def print\_last\_digit(number):

    if number < 0:

        last\_digit = number % -(10)

        print(-(last\_digit), end='')

    else:

        last\_digit = number % 10

        print(last\_digit, end='')

    return abs(last\_digit)

**10-add.py**

#!/usr/bin/python3

def add(a, b):

    return (a + b)

**11-pow.py**

#!/usr/bin/python3

def pow(a, b):

    return (a \*\* b)

**12-fizzbuz.py**

#!/usr/bin/python3

def fizzbuzz():

    for number in range(1, 101):

        if number % 3 == 0 and number % 5 == 0:

            print("FizzBuzz ", end="")

        elif number % 3 == 0:

            print("Fizz ", end="")

        elif number % 5 == 0:

            print("Buzz ", end="")

        else:

            print("{} ".format(number), end="")

**13-insert\_number.c**

#include "lists.h"

/\*\*

 \* insert\_node - Inserts a number into a sorted singly-linked list.

 \* @head: A pointer the head of the linked list.

 \* @number: The number to insert.

 \* Return: 0 If the function fails or pointer to the new node.

 \*/

listint\_t \*insert\_node(listint\_t \*\*head, int number)

{

        listint\_t \*node = \*head, \*new;

        new = malloc(sizeof(listint\_t));

        if (new == NULL)

                return (NULL);

        new->n = number;

        if (node == NULL || node->n >= number)

        {

                new->next = node;

                \*head = new;

                return (new);

        }

        while (node && node->next && node->next->n < number)

                node = node->next;

        new->next = node->next;

        node->next = new;

        return (new);

}

**100-print\_tehbapla.py**

#!/usr/bin/python3

for i in range(ord('z'), ord('a') - 1, -1):

    if i % 2 == 0:

        diff = 0

    else:

        diff = 32

    print('{}'.format(chr(i - diff)), end='')

**101-revove\_char\_at.py**

#!/usr/bin/python3

def remove\_char\_at(str, n):

    if n < 0:

        return (str)

    return (str[:n] + str[n+1:])

**102-magic\_calculation.py**

#!/usr/bin/python3

def magic\_calculation(a, b, c):

    if a < b:

        return (c)

    if c > b:

        return (a + b)

    return (a\*b - c)