Exercise Sheet 1

Task 1.3 (i)

I created the frequency table using the following programm:

```python
import re


def read_file():
    with open('./01-3.txt') as f:
        return f.read()


def count_chars(string: str):
    map_count = {}
    for c in string:
        map_count.setdefault(c, 0)
        map_count[c] += 1
    return map_count


def absolute_char_count(chars):
    result = 0
    for char in chars:
        result += char[1]
    return result


def main():
    string = read_file()
    item_set = count_chars(string).items()
    filtered_list = list(filter(lambda x: re.match(r'[A-Z]', x[0]), item_set))
    filtered_list.sort(key=lambda x: x[1], reverse=True)
    abs_count = absolute_char_count(filtered_list)
    print('Frequency table:')
    print('Letter | Frequency (absolute) | Frequency (percentage)')
    for letter in filtered_list:
        print(letter[0] + ' | ' + str(letter[1]) + ' | ' + str(letter[1] / abs_count))
```

I read in the file, created a map containing each char in the text and counted it. After that I filtered it such that the list I analyze only contains characters from the alphabet.

The resulting table looks like this:

```
Frequency table (sorted in ascending order):
Letter | Frequency (absolute) | Frequency (percentage)
T | 182 | 0.13561847988077497
X | 125 | 0.09314456035767511
D | 118 | 0.08792846497764531
F | 98 | 0.07302533532041729
N | 95 | 0.07078986587183309
Y | 84 | 0.06259314456035768
K | 78 | 0.05812220566318927
H | 73 | 0.05439642324888227
G | 72 | 0.05365126676602087
E | 70 | 0.05216095380029806
J | 54 | 0.040238450074515646
M | 39 | 0.029061102831594635
U | 37 | 0.027570789865871834
W | 34 | 0.02533532041728763
Q | 30 | 0.022354694485842028
C | 30 | 0.022354694485842028
R | 29 | 0.021609538002980627
A | 27 | 0.020119225037257823
I | 27 | 0.020119225037257823
P | 21 | 0.01564828614008942
V | 10 | 0.007451564828614009
O | 6 | 0.004470938897168405
B | 2 | 0.0014903129657228018
S | 1 | 0.0007451564828614009
```

Task 1.3 (ii)

I deciphered the text via the following method: I took the frequency table and then substituted the first three letters in the table with the three most frequent characters of the english language I obtained from Wikipedia. After that I searched for small words like "at", "the", "or" and "and". These generally prove to be good starting points to rebuild the key. After finding some of these and replacing the letters in the cipher text with the corresponding letters of the alphabet, I went ahead and tried to substitute some letters in the cipher text with letters from the Wikipedia frequency table. After that I could just read the text and substitute any letters that where "off" in the words with the correct letter, resulting in the following deciphered text:

```
FAR OUT IN THE UNCHARTED BACKWATERS OF THE UNFASHIONABLE  END  OF
THE  WESTERN  SPIRAL  ARM  OF  THE GALAYY LIES A SMALL UNREGARDED
YELLOW SUN.

ORBITING THIS AT A DISTANCE OF ROUGHLY NINETY-TWO  MILLION  MILES
IS  AN  UTTERLY INSIGNIFICANT LITTLE BLUE GREEN PLANET WHOSE APE-
DESCENDED LIFE FORMS ARE SO AMAZINGLY PRIMITIVE THAT  THEY  STILL
THINK DIGITAL WATCHES ARE A PRETTY NEAT IDEA.

THIS PLANET HAS - OR RATHER HAD - A PROBLEM, WHICH WAS THIS: MOST
OF  THE  PEOPLE  ON  IT WERE UNHAPPY FOR PRETTY MUCH OF THE TIME.
MANY SOLUTIONS WERE SUGGESTED FOR THIS PROBLEM, BUT MOST OF THESE
WERE  LARGELY  CONCERNED WITH THE MOVEMENTS OF SMALL GREEN PIECES
OF PAPER, WHICH IS ODD BECAUSE ON THE WHOLE IT WASN'T  THE  SMALL
GREEN PIECES OF PAPER THAT WERE UNHAPPY.

AND SO THE PROBLEM REMAINED; LOTS OF THE PEOPLE  WERE  MEAN,  AND
MOST OF THEM WERE MISERABLE, EVEN THE ONES WITH DIGITAL WATCHES.

MANY WERE INCREASINGLY OF THE OPINION THAT THEY'D ALL MADE A  BIG
MISTAKE  IN  COMING  DOWN  FROM THE TREES IN THE FIRST PLACE. AND
SOME SAID THAT EVEN THE TREES HAD BEEN A BAD MOVE,  AND  THAT  NO
ONE SHOULD EVER HAVE LEFT THE OCEANS.

AND THEN, ONE THURSDAY, NEARLY TWO THOUSAND YEARS AFTER  ONE  MAN
HAD  BEEN NAILED TO A TREE FOR SAYING HOW GREAT IT WOULD BE TO BE
NICE TO PEOPLE FOR A CHANGE, ONE GIRL SITTING ON  HER  OWN  IN  A
SMALL  CAFE  IN  RICKMANSWORTH SUDDENLY REALIZED WHAT IT WAS THAT
HAD BEEN GOING WRONG ALL THIS TIME, AND SHE FINALLY KNEW HOW  THE
WORLD  COULD  BE  MADE  A  GOOD AND HAPPY PLACE. THIS TIME IT WAS
RIGHT, IT WOULD WORK, AND NO ONE WOULD  HAVE  TO  GET  NAILED  TO
ANYTHING.

SADLY, HOWEVER, BEFORE SHE COULD GET TO A PHONE  TO  TELL  ANYONE-
ABOUT  IT,  A  TERRIBLY STUPID CATASTROPHE OCCURRED, AND THE IDEA
WAS LOST FOREVER.
```

The full programm that was used for this can be seen here:

```python
import re



def read_file():
    with open('./01-3.txt') as f:
        return f.read()



def count_chars(string: str):
    map_count = {}
    for c in string:
        map_count.setdefault(c, 0)
        map_count[c] += 1
    return map_count


def absolute_char_count(chars):
    result = 0
    for char in chars:
        result += char[1]
    return result


def main():
    string = read_file()
    item_set = count_chars(string).items()
    filtered_list = list(filter(lambda x: re.match(r'[A-Z]', x[0]), item_set))
    filtered_list.sort(key=lambda x: x[1], reverse=True)
    abs_count = absolute_char_count(filtered_list)
    print('Frequency table:')
    print('Letter | Frequency (absolute) | Frequency (percentage)')
    for letter in filtered_list:
        print(letter[0] + ' | ' + str(letter[1]) + ' | ' + str(letter[1] / abs_count))

    replacement_table = {
        'T': 'E',
        'X': 'T',
        'D': 'A',
        'F': 'O',
        'K': 'H',
        'J': 'D',
        'G': 'S',
        'I': 'Y',
        'Y': 'I',
        'R': 'C',
        'H': 'R',
        'W': 'P',
        'E': 'L',
        'P': 'B',
        'A': 'U',
        'M': 'W',
        'C': 'G',
        'Q': 'F',
        'U': 'M',
        'O': 'K',
        'S': 'Y',
        'B': 'Z',
        'V': 'V'
    }
    print(string.translate(str.maketrans(replacement_table)))
```

```
if __name__ == '__main__':
    main()
```

The key is the "replacement_table" above