Отчёта по лабораторной работе

Шифры простой замены

Шутенко Виктория Михайловна

Содержание

Список иллюстраций

1 Цель работы

Приобрести практические навыки работы с шифрами простой замены.

2 Ход работы

1. В первом задании выполнялось написание шифра Цезаря. Для написания использовался высокоуровневый язык программирования python. Для реализации работы шифра создана функция `def ceasar (text, k)`, которая принимает на входе текст и компонент отвечающий за ключ. С помощью цикла `for` оббегаем весь текст и задаем условия с помощью цикла `if-elif`:

- если i-й элемент является пробелом, то пропускаем его;
- если i-й элемент является заглавным, то переводим его в unicode, делаем сдвиг на k элементов, и возвращаем в ASCII;
- если i-й элемент не является заглавным, то переводим его в unicode, делаем сдвиг на k элементов, и возвращаем в ASCII как и для заглавного.

Далее делаем вывод исходного текста и зашифрованного. В качестве примера кодировалось фраза “heLLo world” с ключом равным 5 (k=5). В результате кодирования получилось “mjQQt btwqi”.

```
def ceasar (text, k):
    result = ""
    for i in text:
        if i == " ":
            result += i
        elif i.isupper():
            i_unicode = ord(i)
```

```
        i_index = ord(i) - ord("A")
        new_index = (i_index+k)%26
        new_unicode = new_index + ord("A")
        new_char=chr(new_unicode)
        result = result + new_char
    elif i != i.isupper():
        i_unicode = ord(i)
        i_index = ord(i) - ord("a")
        new_index = (i_index+k)%26
        new_unicode = new_index + ord("a")
        new_char=chr(new_unicode)
        result = result + new_char
print ("Plain text: " + text)
print ("Encrypted text: " + result)
```

[141]:
```
def ceasar (text, k):
    result = ""
    for i in text:
        if i == " ":
            result += i
        elif i.isupper():
            i_unicode = ord(i)
            i_index = ord(i) - ord("A")
            new_index = (i_index+k)%26
            new_unicode = new_index + ord("A")
            new_char=chr(new_unicode)
            result = result + new_char
        elif i != i.isupper():
            i_unicode = ord(i)
            i_index = ord(i) - ord("a")
            new_index = (i_index+k)%26
            new_unicode = new_index + ord("a")
            new_char=chr(new_unicode)
            result = result + new_char
    print ("Plain text: " + text)
    print ("Encrypted text: " + result)
```

[142]:
```
ceasar("heLLo world", 5)
```

```
Plain text: heLLo world
Encrypted text: mjQQt btwqi
```

Рис. 2.1: Шифр Цезаря

2. Для второго задания осуществлялась реализация шифра Атбаша. Здесь уже предполагается, что ключ - это инверсия алфавита, следовательно:

A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | W | X | Y | Z |
Z | Y | X | W | V | U | T | S | R | Q | P | O | N | M | L | K | J | I | H | G | F | E | D | C | B | A |

Для реализации работы шифра создана функция `atbash (text)`, которая принимает на входе текст и компонент отвечающий за ключ. С помощью цикла `for` оббегаем весь текст и задаем условия с помощью цикла `if-elif`:

- если i-й элемент является пробелом, то пропускаем его;
- если i-й элемент является заглавным, то переводим его в unicode, делаем сдвиг на 25 элементов, и возвращаем в ASCII;
- если i-й элемент не является заглавным, то переводим его в unicode, делаем сдвиг на 25 элементов, и возвращаем в ASCII как и для заглавного.

Далее делаем вывод исходного текста и зашифрованного. В качестве примера кодировалось фраза “heLLo world”, а результат кодирования — “mjQQt btwqi”.

```
def atbash (text):
    result = ""
    for i in text:
        if i == " ":
            result += i
        elif i.isupper():
            i_unicode = ord(i)
            i_index = ord(i) - ord("A")
            new_index = 25-i_index%26
            new_unicode = new_index + ord("A")
            new_char=chr(new_unicode)
            result = result + new_char
```

```
    elif i != i.isupper():
        i_unicode = ord(i)
        i_index = ord(i) - ord("a")
        new_index = 25-i_index%26
        new_unicode = new_index + ord("a")
        new_char=chr(new_unicode)
        result = result + new_char
print ("Plain text: " + text)
print ("Encrypted text: " + result)
```

[143]:
```
def atbash (text):
    result = ""
    for i in text:
        if i == " ":
            result += i
        elif i.isupper():
            i_unicode = ord(i)
            i_index = ord(i) - ord("A")
            new_index = 25-i_index%26
            new_unicode = new_index + ord("A")
            new_char=chr(new_unicode)
            result = result + new_char
        elif i != i.isupper():
            i_unicode = ord(i)
            i_index = ord(i) - ord("a")
            new_index = 25-i_index%26
            new_unicode = new_index + ord("a")
            new_char=chr(new_unicode)
            result = result + new_char
    print ("Plain text: " + text)
    print ("Encrypted text: " + result)
```

[144]:
```
atbash("Hello world")
```

```
Plain text: Hello world
Encrypted text: Svool dliow
```

Рис. 2.2: Шифр Атбаша