# コンピュータサイエンス第一
# 課題３「暗号解読に挑戦」

学籍番号******** ****

**概要**

古典的な暗号化である Caesar 暗号の暗号化、復号化およびその解読のプログラムを作成した。

## 1　実行方法と出力例

ターミナルを開き、kadai3 ディレクトリ内で、以下のコマンドを実行することにより、本プログラムを実行できる。

### 1.1　暗号化

```
$ ruby ango.rb < h0.txt
```

上の実行結果は以下の通りになる。

```
Hhoor ghdu,  I oryh brx!
```

### 1.2　復号化

上の実行結果を crypted.txt に保存したとき、以下のコマンドで復号できる。

```
$ ruby hukugo.rb < crypted.txt
```

```
Hello dear,  I love you!
```

### 1.3　解読

```
$ ruby kaidoku.rb < a1.txt
```

実行結果の文字列が長いため、一部省略して表示する。

63
115
119
(省略)
holmes had been seated for some hours in silence with his long(以下省略)

## 2　暗号解読プログラムの仕組み

英文では、一般に e の文字が最も登場頻度が高いことが知られている。Caesar 暗号は単にアルファベットを一定の幅だけずらしたものであるから、暗号文において最も登場する文字を数え、その文字が e とどれだけずれているか計算することによって、鍵を特定できる。その後は得た鍵を使って暗号化の逆にあたる処理を行うことによって復号している。

## 3　工夫した点

暗号化、復号化で小文字のアルファベットの文字コードを加減する際、変換後の文字コードが小文字の範囲 (97 から 112) に収まっていないと、その外の文字になってしまう不具合が起こる。そのため、シフト処理には剰余を用い、変換後のコードと a のコードとの差が 0 から 25 になるようにしている。

## 4　ソースコード

以下に kaidoku.rb のソースコードを示す。

```
# kaidoku.rb
# 解読プログラム
# 入力: 暗号文の文字列
# 出力: 復号した平文

# 平文を暗号化するサブルーチン
# dec(秘密鍵 k, 暗号文 c) = 平文 m
def dec(k, c)
  code_a = 97                  # 文字 a の文字コード
  kosu = 26                    # 英字アルファベットの数

  leng = c.length              # 文字列の長さ
  a = c.unpack("C*")           # 文字列から文字コードの配列へ変換
  b = Array.new(leng)          # 暗号文（のコード）格納用配列
```

```
  for i in 0..leng-1
    code = a[i]                  # i 文字目のコードを得る
    sa = code - code_a           # 文字 a からの差分
    if 0 <= sa and sa < kosu     # 小文字ならば文字を kの数分だけずらす
      sa = (sa - k) % kosu
    end
    b[i] = sa + code_a
  end
  m = b.pack("C*")               # コードの配列を文字列に直す
  return(m)
end
# サブルーチン dec （終）

##### プログラム本文 #####
# 入力と準備
code_a = 97                  # 文字 a の文字コード
kosu = 26                    # 英字アルファベットの数
angobun = gets.chomp         # 暗号文を入力（注：入力文字列から改行を除去）
leng = angobun.length        # 暗号文の長さを leng に格納
a = angobun.unpack("C*")     # 文字列から文字コードの配列へ変換
hindo = Array.new(kosu, 0) # 頻度配列の準備

# 頻度配列の作成
for i in 0..leng-1
  code = a[i]                    # i 文字目のコードを得る
  sa = code - code_a             # 文字 a からの差分
  if 0 <= sa and sa < kosu       # 小文字なら
    hindo[sa] = hindo[sa] + 1      # 頻度配列の該当する要素をインクリメント
  end
end
puts(hindo)  ## 確認用

# 最頻文字の位置（差分）の確定
max = 0
maxj = -1
for j in 0..(kosu-1)
  if hindo[j] > max
```

```
    max = hindo[j]
    maxj = j
  end
end
puts(max, maxj)  ## 確認用

k = (maxj + 1 - 5) % kosu       # maxj から暗号鍵を求める
hirabun = dec(k, angobun)  # この暗号鍵で平文に変換
puts(hirabun)              # 解読平文を出力
```