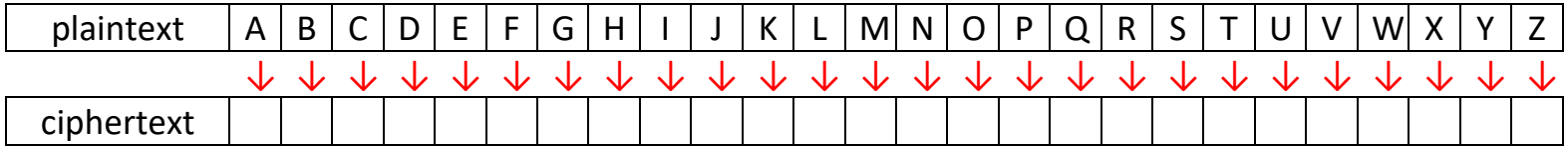
Python Caesar Cipher with Tkinter

plus Encryption and Decryption Practice Worksheets

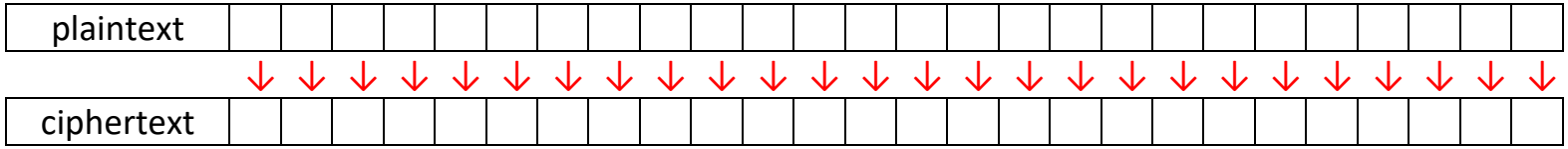
Compucademy

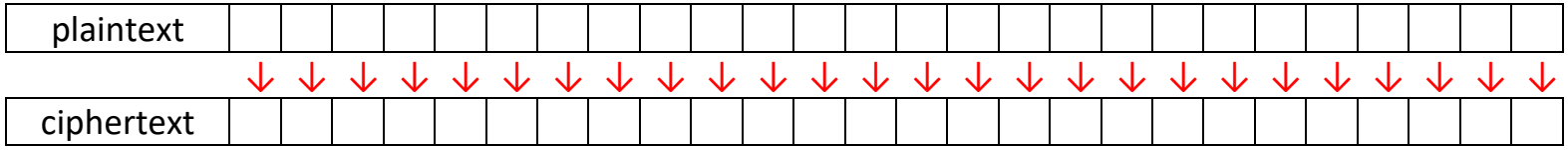
Caesar Cipher Practice Sheet - Encryption

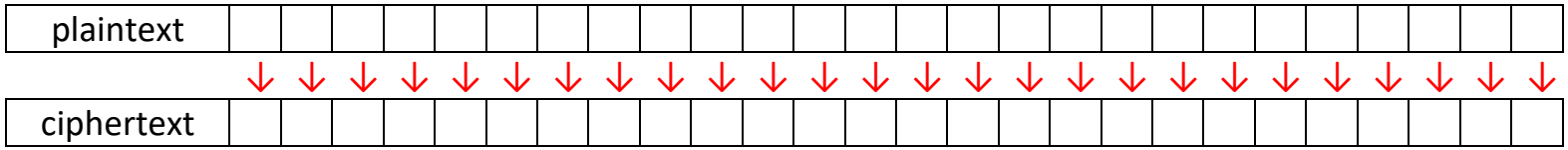
Key:

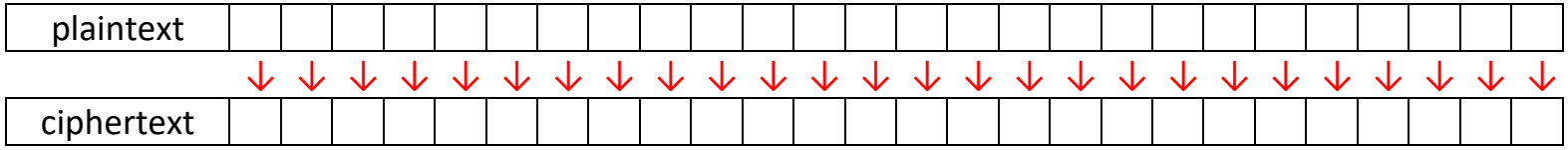


Message/s:



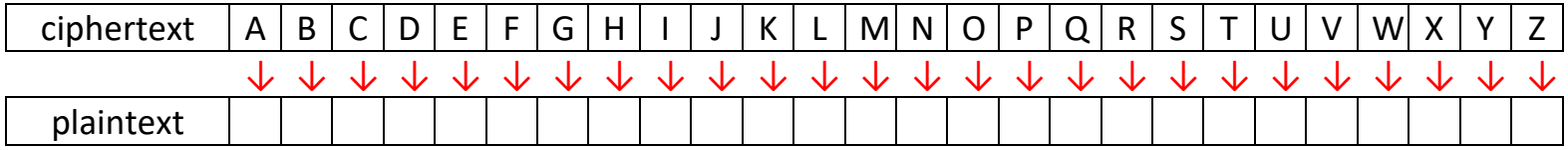




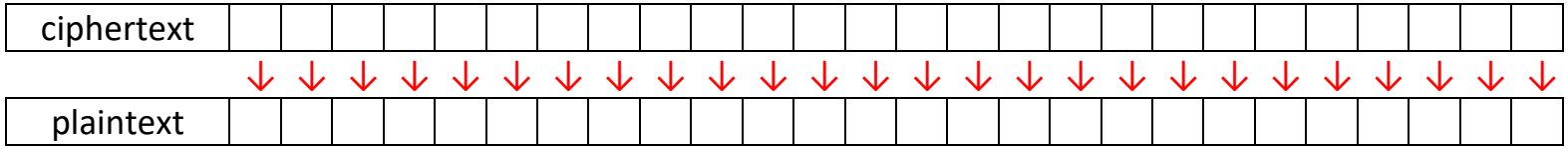


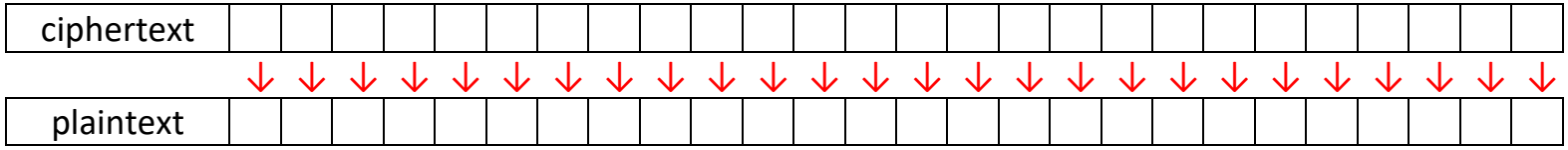
Caesar Cipher Practice Sheet - Decryption

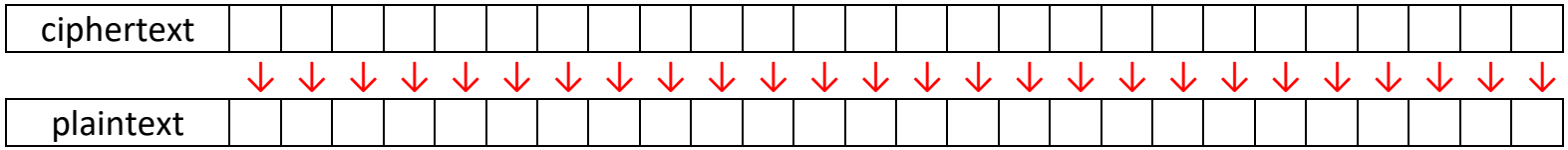
Key:

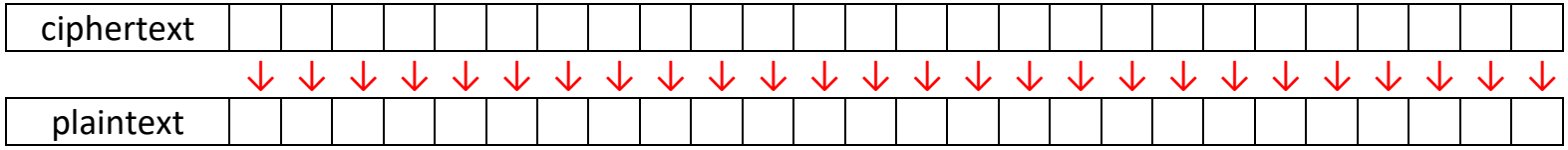


Message/s:









Python Caesar Cipher with Tkinter Code Listing

```
import tkinter as tk

FONT = ("calbri", 20, "bold")


class CaesarCipherGUI:
    def __init__(self, master):
        master.title("Caesar Cipher GUI")
        self.plaintext = tk.StringVar(master, value="")
        self.ciphertext = tk.StringVar(master, value="")
        self.key = tk.IntVar(master)

        # Plaintext controls
        self.plain_label = tk.Label(master, text="Plaintext", fg="green", font=FONT).grid(row=0, column=0)
        self.plain_entry = tk.Entry(master,
                                    textvariable=self.plaintext, width=50, font=FONT)
        self.plain_entry.grid(row=0, column=1, padx=20)
        self.encrypt_button = tk.Button(master, text="Encrypt",
                                        command=lambda: self.encrypt_callback(), font=FONT).grid(row=0, column=2)
        self.plain_clear = tk.Button(master, text="Clear",
                                     command=lambda: self.clear('plain'), font=FONT).grid(row=0, column=3)

        # Key controls
        self.key_label = tk.Label(master, text="Key", font=FONT).grid(row=1, column=0)
        self.key_entry = tk.Entry(master, textvariable=self.key, width=10, font=FONT).grid(row=1, column=1,
                                                                                            sticky=tk.W, padx=20)

        # Ciphertext controls
        self.cipher_label = tk.Label(master, text="Ciphertext", fg="red", font=FONT).grid(row=2, column=0)
        self.cipher_entry = tk.Entry(master,
                                     textvariable=self.ciphertext, width=50, font=FONT)
        self.cipher_entry.grid(row=2, column=1, padx=20)
        self.decrypt_button = tk.Button(master, text="Decrypt",
                                        command=lambda: self.decrypt_callback(), font=FONT).grid(row=2, column=2)
        self.cipher_clear = tk.Button(master, text="Clear",
                                      command=lambda: self.clear('cipher'), font=FONT).grid(row=2, column=3)

    def clear(self, str_val):
        if str_val == 'cipher':
            self.cipher_entry.delete(0, 'end')
        elif str_val == 'plain':
            self.plain_entry.delete(0, 'end')

    def get_key(self):
        try:
            key_val = self.key.get()
            return key_val
        except tk.TclError:
            pass

    def encrypt_callback(self):
        key = self.get_key()
        ciphertext = encrypt(self.plain_entry.get(), key)
        self.cipher_entry.delete(0, tk.END)
        self.cipher_entry.insert(0, ciphertext)

    def decrypt_callback(self):
        key = self.get_key()
        plaintext = decrypt(self.cipher_entry.get(), key)
        self.plain_entry.delete(0, tk.END)
        self.plain_entry.insert(0, plaintext)


def encrypt(plaintext, key):
    ciphertext = ""
    for char in plaintext.upper():
        if char.isalpha():
            ciphertext += chr((ord(char) + key - 65) % 26 + 65)
        else:
            ciphertext += char
```

```
        return ciphertext


    def decrypt(ciphertext, key):
        plaintext = ""
        for char in ciphertext.upper():
            if char.isalpha():
                plaintext += chr((ord(char) - key - 65) % 26 + 65)
            else:
                plaintext += char
        return plaintext


    if __name__ == "__main__":
        root = tk.Tk()
        caesar = CaesarCipherGUI(root)
        root.mainloop()
```

Printable ASCII Characters

Dec	Hex	Binary	Character	Description
32	20	00100000	**Space**	**space**
33	21	00100001	**!**	exclamation mark
34	22	00100010	**"**	double quote
35	23	00100011	**#**	number
36	24	00100100	**$**	dollar
37	25	00100101	**%**	percent
38	26	00100110	**&**	ampersand
39	27	00100111	**'**	single quote
40	28	00101000	**(**	left parenthesis
41	29	00101001	**)**	right parenthesis
42	2A	00101010	*	asterisk
43	2B	00101011	**+**	plus
44	2C	00101100	**,**	comma
45	2D	00101101	**-**	minus
46	2E	00101110	**.**	period
47	2F	00101111	**/**	slash
48	30	00110000	**0**	zero
49	31	00110001	**1**	one
50	32	00110010	**2**	two
51	33	00110011	**3**	three
52	34	00110100	**4**	four
53	35	00110101	**5**	five
54	36	00110110	**6**	six

Dec	Hex	Binary	Character	Description
80	50	01010000	**P**	
81	51	01010001	**Q**	
82	52	01010010	**R**	
83	53	01010011	**S**	
84	54	01010100	**T**	
85	55	01010101	**U**	
86	56	01010110	**V**	
87	57	01010111	**W**	
88	58	01011000	**X**	
89	59	01011001	**Y**	
90	5A	01011010	**Z**	
91	5B	01011011	**[**	left square bracket
92	5C	01011100	****	backslash
93	5D	01011101	**]**	right square bracket
94	5E	01011110	**^**	caret / circumflex
95	5F	01011111	**_**	underscore
96	60	01100000	**`**	grave / accent
97	61	01100001	**a**	
98	62	01100010	**b**	
99	63	01100011	**c**	
100	64	01100100	**d**	
101	65	01100101	**e**	
102	66	01100110	**f**	

Dec	Hex	Binary	Character	Description
55	37	00110111	**7**	seven
56	38	00111000	**8**	eight
57	39	00111001	**9**	nine
58	3A	00111010	**:**	colon
59	3B	00111011	**;**	semicolon
60	3C	00111100	**<**	less than
61	3D	00111101	**=**	equality sign
62	3E	00111110	**>**	greater than
63	3F	00111111	**?**	question mark
64	40	01000000	**@**	at sign
65	41	01000001	**A**	
66	42	01000010	**B**	
67	43	01000011	**C**	
68	44	01000100	**D**	
69	45	01000101	**E**	
70	46	01000110	**F**	
71	47	01000111	**G**	
72	48	01001000	**H**	
73	49	01001001	**I**	
74	4A	01001010	**J**	
75	4B	01001011	**K**	
76	4C	01001100	**L**	
77	4D	01001101	**M**	
78	4E	01001110	**N**	
79	4F	01001111	**O**	

Dec	Hex	Binary	Character	Description
103	67	01100111	**g**	
104	68	01101000	**h**	
105	69	01101001	**i**	
106	6A	01101010	**j**	
107	6B	01101011	**k**	
108	6C	01101100	**l**	
109	6D	01101101	**m**	
110	6E	01101110	**n**	
111	6F	01101111	**o**	
112	70	01110000	**p**	
113	71	01110001	**q**	
114	72	01110010	**r**	
115	73	01110011	**s**	
116	74	01110100	**t**	
117	75	01110101	**u**	
118	76	01110110	**v**	
119	77	01110111	**w**	
120	78	01111000	**x**	
121	79	01111001	**y**	
122	7A	01111010	**z**	
123	7B	01111011	**{**	left curly bracket
124	7C	01111100	**|**	vertical bar
125	7D	01111101	**}**	right curly bracket
126	7E	01111110	**~**	tilde
127	7F	01111111	DEL	delete