Practical 2\Practical.py

```python
#####################################################################
                        #Practical 2A
#####################################################################
def encrypt(inputText: str, key: int) -> str:
    matrix: list[list[chr]] = [list(" ") * len(inputText) for _ in range(key)]
    i: int = 0
    flag: bool = True
    for j, ch in enumerate(inputText):
        matrix[i][j] = ch
        if flag:
            i += 1
        else:
            i -= 1
        if i == key - 1 or i == 0:
            flag = not (flag)

    for row in matrix:
        print(*row)

    result: str = ""
    for row in matrix:
        result += "".join(row).replace(" ", "").upper()

    return result


def decrypt(inputText: str, key: int) -> str:
    matrix: list[list[chr]] = [list(" ") * len(inputText) for _ in range(key)]
    result: list[str] = [" " for _ in range(len(inputText))]

    ch: int = 0
    for row in range(key):
        i: int = 0
        flag: bool = True
        for x in range(len(inputText)):
            if i == row:
                matrix[row][x] = inputText[ch]
                result[x] = inputText[ch].lower()
                ch += 1
            if flag:
                i += 1
            else:
                i -= 1
            if i == key - 1 or i == 0:
                flag = not (flag)

    for row in matrix:
        print(*row)

    return "".join(result).replace(" ", "")


```

```python
def main() -> None:
    print(f'{'start':-^40}')
    inputText: str = input("Enter your String :").strip().replace(" ", "")
    key: int = int(input("Enter Key: "))

    print("-" * 40)
    if inputText.islower():
        res = encrypt(inputText, key)
        print("-" * 40)
        print("Encrypt :", res)
    else:
        res = decrypt(inputText, key)
        print("-" * 40)
        print("Decrypt :", res)
    print(f'{'end':-^40}')


if __name__ == "__main__":
    main()
######################################################################
                        #Practical 2B
######################################################################
def simpleColumnar(inputText: str, column: int):
    row: int = (len(inputText) / column).__ceil__()
    inputText += "*" * (row * column - len(inputText))

    if inputText.islower():
        matrix: list[str] = [
            inputText[i : i + column] for i in range(0, len(inputText), column)
        ]

        encrypt: str = ""
        for text in matrix:
            print(*text, sep=" ")

        for i in range(column):
            for text in matrix:
                encrypt += text[i].upper()

        print('-'*40)
        print("Encrypted:", encrypt)
    else:
        matrix: list[list[chr]] = [ list(" ") * column for r in range(row)]

        i: int = 0
        for c in range(column):
            for r in range(row):
                matrix[r][c] = inputText[i]
                i+=1

        for text in matrix:
            print(*text, sep=" ")

        decrypt: str = ""
```

```
        for text in matrix:
            decrypt += ''.join(text).lower()

        print('-'*40)
        print("Decrypted:", decrypt)


def main() -> None:
    while True:
        print(f'{'start':-^40}')
        inputText: str = input("Enter your String :").replace(" ", "")
        column: int = int(input("Enter Column: "))
        print("-" * 40)
        simpleColumnar(inputText=inputText, column=column)
        print(f'{'end':-^40}')

if __name__ == "__main__":
    main()
#####################################################################
                        #Practical 2C
#####################################################################
def main() -> None:
    print(f'{'start':-^40}')
    enkey: list[int] = map(int, input("Enter Your key: ").strip().split(" "))


    enkey: list[list[int]] = list(map(list, enumerate(enkey, start=1)))
    dekey: list[list[int]] = sorted(enkey, key=lambda x: x[1])

    enkey: list[int] = [x[1] for x in enkey]
    dekey: list[int] = [x[0] for x in dekey]

    print('-' * 40)
    print(f'Encryption Key: ',*enkey)

    print(f'Decryption Key: ',*dekey)
    print(f'{'end':-^40}')

if __name__ == '__main__':
    main()
#####################################################################
                        #Practical 2D
#####################################################################
def hybridTransposition(inputText: str, enkey: list[list[int]]) -> None:
    enkey: list[list[int]] = list(map(list, enumerate(enkey, start=1)))
    dekey: list[list[int]] = sorted(enkey, key=lambda x: x[1])

    col: int = len(enkey)
    row: int = (len(inputText) / col).__ceil__()
    inputText += "*" * ((row * col) - len(inputText))

    enkey: list[int] = [x[1] for x in enkey]
    dekey: list[int] = [x[0] for x in dekey]

```

```
    result = ""

    if inputText.islower():
        print(f'Encrypted Key: ', enkey)
        print('-'*40)

        matrix: list[list[chr]] = [
            inputText[x : x + col] for x in range(0, len(inputText), col)
        ]
        for text in matrix:
            print(*text, sep=" ")
        print('-'*40)

        for e in enkey:
            for l in matrix:
                result += l[e - 1].upper()

        print(f'Encrypted: ',result)
    else:
        print(f'Decrypted Key: ', dekey)
        print('-'*40)

        matrix: list[list[chr]] = [ list(" ") * col for r in range(row)]

        i: int = 0
        for c in range(col):
            for r in range(row):
                matrix[r][c] = inputText[i]
                i+=1

        for text in matrix:
            print(*text, sep=" ")
        print('-'*40)

        for l in matrix:
            for d in dekey:
                result += l[d - 1].lower()

        print(f'Decrypted: ',result)


def main() -> None:
    print(f'{'start':-^40}')
    inputText: str = input("Enter your String :").strip().replace(" ", "")
    enkey: list[int] = map(int, input("Enter Your key: ").strip().split(" "))
    print('-'*40)
    hybridTransposition(inputText, enkey)
    print(f'{'end':-^40}')


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