Practical 4\4.py

```python
def splitWord(word: str) -> tuple:
    while len(word) % 2:
        word += '0'
    n: int = len(word)
    return word[:n // 2], word[n // 2:]

def combineWord(word1: str, word2: str) -> str:
    return word1 + word2

def swapWord(word: str) -> str:
    while len(word) % 2:
        word += '0'
    n: int = len(word)
    return word[n // 2:] + word[:n // 2]

def leftShift(word: str, k: int) -> str:
    return word[k:] + word[:k]

def rightShift(word: str, k: int) -> str:
    return word[-k:] + word[:-k]

def mapPBox(word: str, PTable: list[int]) -> str:
    result = ['0'] * len(PTable)
    for i, v in enumerate(PTable):
        result[i] = word[v]
    return "".join(result)

def mapSBox(word: str, SBox: dict[str, str]) -> str:
    chunk_size = len(next(iter(SBox.keys())))
    result = []
    for i in range(0, len(word), chunk_size):
        chunk = word[i:i + chunk_size]
        result.append(SBox.get(chunk, "0000"))
    return "".join(result)

def feistelRound(left: str, right: str, key: str, PTable: list[int], SBox: dict[str, str]) -> tuple:
    expanded_right = mapPBox(right, PTable)
    xor_result = bin(int(expanded_right, 2) ^ int(key, 2))[2:].zfill(len(expanded_right))
    sbox_result = mapSBox(xor_result, SBox)
    new_right = bin(int(left, 2) ^ int(sbox_result, 2))[2:].zfill(len(left))

    print(f'\tExpanded right: {expanded_right:^20}')
    print(f'\tXor Result     : {xor_result:^20}')
    print(f'\tSbox Result    : {sbox_result:^20}')
    print(f'\tNew Right      : {new_right:^20}')

    return right, new_right

def feistelCipher(word: str, keys: list[str], PTable: list[int], SBox: dict[str, str], rounds: int) -> str:
    left, right = splitWord(word)

    print(f'{left=:^10} {right=:^10}')


    for i in range(rounds):
      print(f'='*40)
      print(f'Round: {i+1}')
      print(f'='*40)
      print(f'Before: {left=:^20} {right=:^20}')
      left, right = feistelRound(left, right, keys[i], PTable, SBox)
      print(f'After:  {left=:^20} {right=:^20}')

      print(f'='*40)


    return combineWord(right, left)

def main():

    word = bin(int(input("Enter Number (0 to 65535):")))[2:].zfill(16)


    keys = ["10101011", "11001010", "00111000", "01010101"]

    PTable = [3, 0, 7, 2, 1, 6, 5, 4]

    SBox = {
        "0000": "1110", "0001": "0100", "0010": "1101", "0011": "0001",
        "0100": "0010", "0101": "1111", "0110": "1011", "0111": "1000",
        "1000": "0011", "1001": "1010", "1010": "0110", "1011": "1100",
        "1100": "0101", "1101": "1001", "1110": "0000", "1111": "0111"
    }

    rounds = 2

    encrypted_word = feistelCipher(word, keys, PTable, SBox, rounds)
    print(f"Encrypted word: {encrypted_word}")
    print(f"Decimal       : {int(encrypted_word,2)}")

if __name__ == '__main__':
    main()


####################OUTPUT################################
Enter Number (0 to 65535):563
left= 00000010  right= 00110011
========================================
Round: 1
========================================
Before: left=      00000010       right=      00110011
        Expanded right:       10110100
        Xor Result    :       00011111
        Sbox Result   :       01000111
        New Right     :       01000101
After:  left=      00110011       right=      01000101
========================================
========================================
Round: 2
========================================
Before: left=      00110011       right=      01000101
        Expanded right:       00101010
        Xor Result    :       11100000
        Sbox Result   :       00001110
        New Right     :       00111101
After:  left=      01000101       right=      00111101
========================================
Encrypted word: 0011110101000101
Decimal       : 15685
```