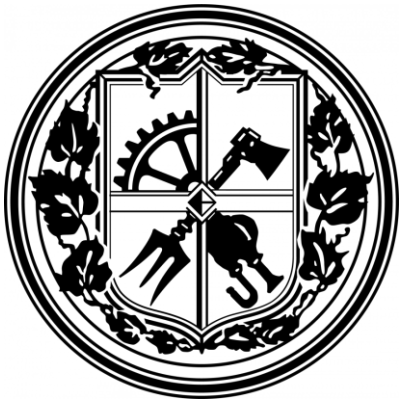

НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ «КИЇВСЬКИЙ ПОЛІТИХНІЧНИЙ ІНСТИТУТ ІМ.ІГОРЯ СІКОРСЬКОГО»

ФІЗИКО-ТЕХНІЧНИЙ ІНСТИТУТ

Комп’ютерний практикум №1

“Рекурсія”

Варіант 5

Виконав: Жуковін Олесандр ФІ-21

6. Рекурсивно обчислити суму цифр заданого натурального числа.

```python
# 6. Рекурсивно обчислити суму цифр заданого натурального числа.
def digit_sum(num, ind=0):
    if ind == len(str(num)):
        return 0
    return int(str(num)[ind]) + digit_sum(num, ind+1)


print(digit_sum(25123))
print(digit_sum(123))
print(digit_sum(298))
```

```
13
6
19
```

14. Реалізувати алгоритм для розв’язання задачі «Ханойські вежі».Виписати послідовність ходів для перекладання n дисків вежі (n = 2; 3; 4; 5 дисків, використати онлайн гру).

```python
# 14. Реалізувати алгоритм для розв’язання задачі «Ханойські вежі».
def hanoi_tower(n, count=0, moves=0, **kwargs):
    kwargs['b'].append(kwargs['a'].pop(-1))
    print(kwargs, moves)
    moves += 1
    kwargs['c'].append(kwargs['a'].pop(-1))
    print(kwargs, moves)
    moves += 1
    kwargs['c'].append(kwargs['b'].pop(-1))
    print(kwargs, moves)
    moves += 1
    if len(kwargs['c']) == n:
        return kwargs
    if count == 0 or count == 3 or count == 6 or count == 9:
        kwargs['b'].append(kwargs['a'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['b'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['a'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['a'].append(kwargs['b'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['c'].append(kwargs['b'].pop(-1))
        print(kwargs, moves)
        moves += 1
        return hanoi_tower(n, count+1, moves, **kwargs)
```

```
        kwargs['c'].append(kwargs['b'].pop(-1))
        print(kwargs, moves)
        moves += 1
        if count == 11:
            return hanoi_tower(n, 0, moves, **kwargs)
        else:
            return hanoi_tower(n, count+1, moves, **kwargs)
    elif count == 5 or count == 7:
        kwargs['a'].append(kwargs['b'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['b'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['a'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['a'].append(kwargs['b'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['b'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        return hanoi_tower(n, count+1, moves, **kwargs)


print(hanoi_tower(4, a=['4', '3', '2', '1'], b=[], c=[]))
```

```
    elif count == 1 or count == 4 or count == 10:
        kwargs['b'].append(kwargs['a'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['b'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['a'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['a'].append(kwargs['b'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['b'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        return hanoi_tower(n, count+1, moves, **kwargs)
    elif count == 2 or count == 8 or count == 11:
        kwargs['a'].append(kwargs['b'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['b'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['a'].append(kwargs['c'].pop(-1))
        print(kwargs, moves)
        moves += 1
        kwargs['a'].append(kwargs['b'].pop(-1))
        print(kwargs, moves)
        moves += 1
```

```
{'a': ['4', '3', '2'], 'b': ['1'], 'c': []} 0
{'a': ['4', '3'], 'b': ['1'], 'c': ['2']} 1
{'a': ['4', '3'], 'b': [], 'c': ['2', '1']} 2
{'a': ['4'], 'b': ['3'], 'c': ['2', '1']} 3
{'a': ['4'], 'b': ['3', '1'], 'c': ['2']} 4
{'a': ['4', '2'], 'b': ['3', '1'], 'c': []} 5
{'a': ['4', '2', '1'], 'b': ['3'], 'c': []} 6
{'a': ['4', '2', '1'], 'b': [], 'c': ['3']} 7
{'a': ['4', '2'], 'b': ['1'], 'c': ['3']} 8
{'a': ['4'], 'b': ['1'], 'c': ['3', '2']} 9
{'a': ['4'], 'b': [], 'c': ['3', '2', '1']} 10
{'a': [], 'b': ['4'], 'c': ['3', '2', '1']} 11
{'a': [], 'b': ['4', '1'], 'c': ['3', '2']} 12
{'a': ['2'], 'b': ['4', '1'], 'c': ['3']} 13
{'a': ['2', '1'], 'b': ['4'], 'c': ['3']} 14
{'a': ['2', '1'], 'b': ['4', '3'], 'c': []} 15
{'a': ['2'], 'b': ['4', '3', '1'], 'c': []} 16
{'a': [], 'b': ['4', '3', '1'], 'c': ['2']} 17
{'a': [], 'b': ['4', '3'], 'c': ['2', '1']} 18
{'a': ['3'], 'b': ['4'], 'c': ['2', '1']} 19
{'a': ['3'], 'b': ['4', '1'], 'c': ['2']} 20
{'a': ['3', '2'], 'b': ['4', '1'], 'c': []} 21
{'a': ['3', '2', '1'], 'b': ['4'], 'c': []} 22
{'a': ['3', '2', '1'], 'b': [], 'c': ['4']} 23
{'a': ['3', '2'], 'b': ['1'], 'c': ['4']} 24
{'a': ['3'], 'b': ['1'], 'c': ['4', '2']} 25
{'a': ['3'], 'b': [], 'c': ['4', '2', '1']} 26
{'a': [], 'b': ['3'], 'c': ['4', '2', '1']} 27
{'a': [], 'b': ['3', '1'], 'c': ['4', '2']} 28
{'a': ['2'], 'b': ['3', '1'], 'c': ['4']} 29
{'a': ['2', '1'], 'b': ['3'], 'c': ['4']} 30
{'a': ['2', '1'], 'b': [], 'c': ['4', '3']} 31
{'a': ['2'], 'b': ['1'], 'c': ['4', '3']} 32
{'a': [], 'b': ['1'], 'c': ['4', '3', '2']} 33
{'a': [], 'b': [], 'c': ['4', '3', '2', '1']} 34
{'a': [], 'b': [], 'c': ['4', '3', '2', '1']}
```