Студента группы ИС-23 Воронкова Е.В.

Практическое занятие №3

Задание №1

Тема: Составление программ ветвящейся структуры в IDE PyCharm Community.

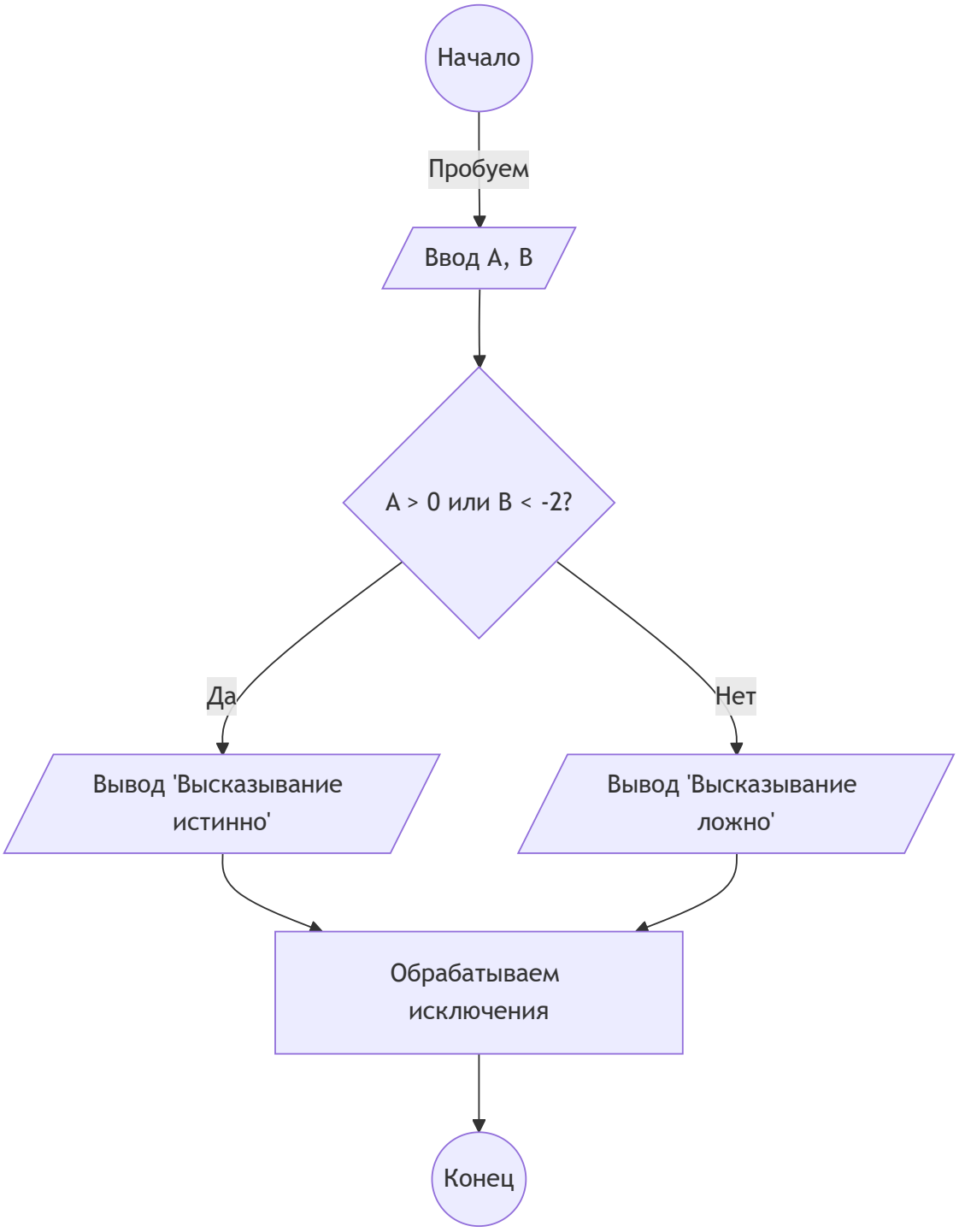
Цель: Закрепить усвоенные знания, понятия, алгоритмы, основные принципы составления программ, приобрести навыки составление программ ветвящейся структуры в IDE PyCharm Community.

Постановка задачи:

Даны два целых числа: A, B. Проверить истинность высказывания: «Справедливы неравенства A > 0 или B < -2».

Тип алгоритма: линейный с ветвлением.

Блок-схема алгоритма:



Текст программы:

```
try:
    a = int(input("Введите число A: "))
    b = int(input("Введите число B: "))
    if a > 0 or b < -2:
        print("Высказывание истинно")
    else:
```

```
        print("Высказывание ложно")
except ValueError:
    print("ОШИБКА! Введены некорректные данные.")
```

Протокол работы программы (примеры):

```
Введите число A: 1
Введите число B: -1
Высказывание истинно

Введите число A: -3
Введите число B: -3
Высказывание истинно

Введите число A: -1
Введите число B: 0
Высказывание ложно

Введите число A: 1
Введите число B: -10
Высказывание истинно


Введите число A: abc
ОШИБКА! Введены некорректные данные.
```

Вывод:

В ходе выполнения практического задания были закреплены навыки работы с логическим оператором "или" для проверки составных условий, а также обработка исключений для обеспечения корректной работы программы при вводе некорректных данных. Блок-схема дополнена обработкой исключений и экранированием текста. В коде программы уточнен тип отлавливаемого исключения для большей специфичности.

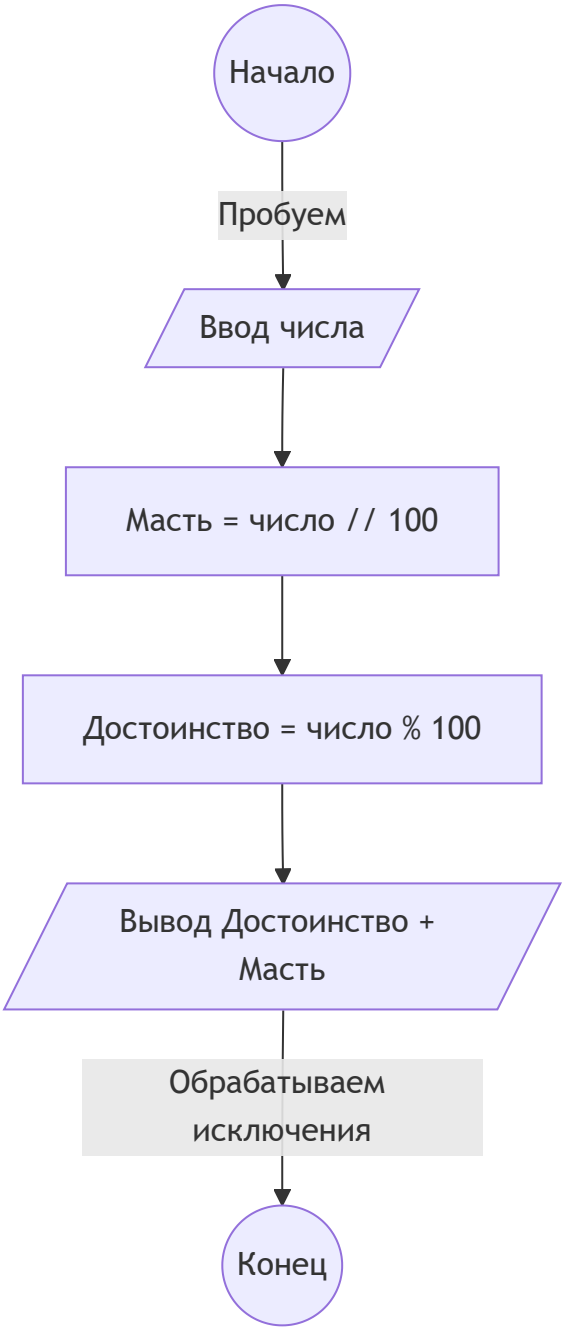
Задание №2

Постановка задачи:

Мастям игральных карт присвоены порядковые номера: 1 – пики, 2 – трефы, 3 – бубны, 4 – червы. Достоинству карт, старших десятки, присвоены номера: 11 – валет, 12 – дама, 13 – король, 14 – туз. Дано трехзначное число, в котором первая цифра указывает на масть, а вторые две на достоинство карты. Вывести соответствующее название карты вида «дама червей», «туз треф» и т.п.

Тип алгоритма: линейный.

Блок-схема алгоритма:



Текст программы:

```
suits = {
    1: "пики",
    2: "трефы",
    3: "бубны",
    4: "червы"
}

dignities = {
    11: "Валет",
```

```
    12: "Дама",
    13: "Король",
    14: "Туз"
}

try:
    num = int(input("Введите трехзначное число: "))
    suit = num // 100
    dignity = num % 100
    print(dignities[dignity], suits[suit])
except ValueError:
    print("ОШИБКА: такой карты нет!")
```

Протокол работы программы (примеры):

```
Введите трехзначное число: 111
Валет пики

Введите трехзначное число: 414
Туз червы

Введите трехзначное число: 212
Дама трефы

Введите трехзначное число: 313
Король бубны

Введите трехзначное число: 115
ОШИБКА: такой карты нет!

Введите трехзначное число: 514
ОШИБКА: такой карты нет!

Введите трехзначное число: abc
ОШИБКА: такой карты нет!
```

Вывод:

В ходе выполнения практического задания были закреплены навыки работы со словарями для хранения информации о мастях и достоинствах карт. Особое внимание уделено обработке исключительных ситуаций. Блок `try...except` теперь обрабатывает как ошибки ввода (`ValueError`), так и ошибки доступа к несуществующим ключам словаря (`KeyError`). Блок-схема отражает эту логику. В отличие от предыдущего варианта, здесь нет явной проверки `if suit in suits and dignity in dignities`, так как ошибки `KeyError` обрабатываются внутри блока `try...except`. Это делает код немного компактнее.