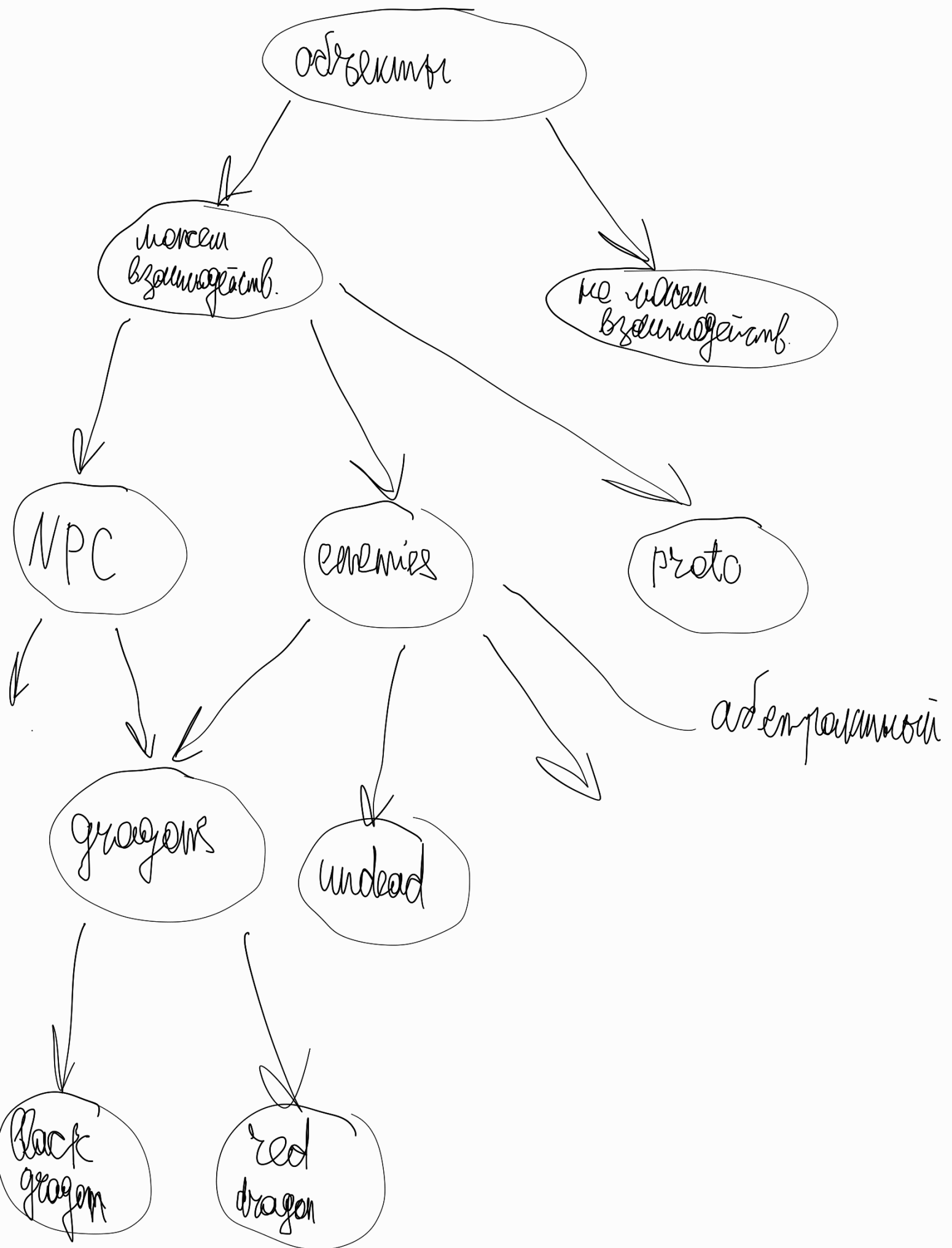


Маскирование,
обработка исключений
Наследование задает отношение
родитель-потомок

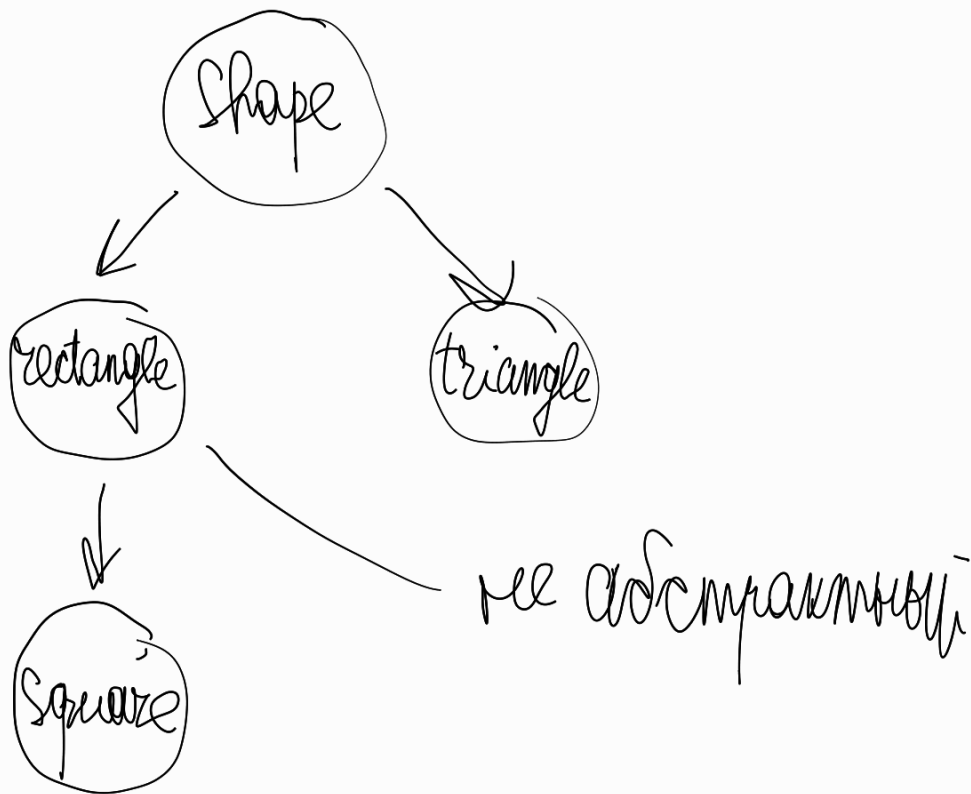
Потомок полностью наследует
и интерфейс и реализацию, но он
может их дополнять и переопределять.

Позволяет структурировать логику используемых
сущностей, уменьшает дублирование кода.

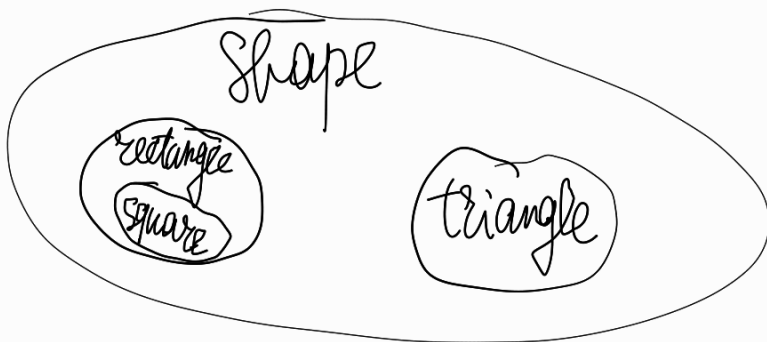
у одного родителя может быть несколько
потомков, у одного потомка несколько
родителей, но порядок родителей имеет
значение.



Класс, создаваемый только для логики
называется абстрактным



С точки зрения программирования, потомок
расширяет предка, но с точки зрения
логики потомок должен уточнять предка,
являться его представителем.



```
class Base:
```

```
    pass
```

```
class Derived(Base):
```

```
    ...
```

```
class A:
```

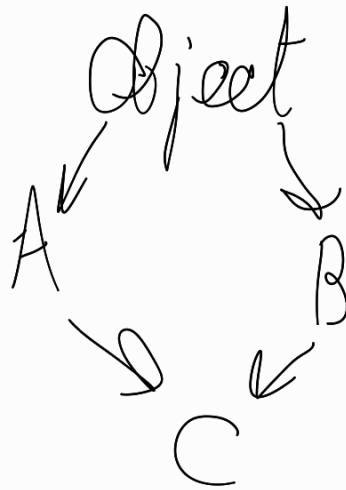
```
    ...
```

```
class B:
```

```
    ...
```

```
class C(A,B):
```

```
    ...
```



`SUPER()` - позволяет обратиться к методам предков.

Не любое исключение может завершить работу программы.

```

BaseException
├── BaseExceptionGroup
├── GeneratorExit
├── KeyboardInterrupt
├── SystemExit
└── Exception
    ├── ArithmeticError
    │   ├── FloatingPointError
    │   ├── OverflowError
    │   └── ZeroDivisionError
    ├── AssertionError
    ├── AttributeError
    ├── BufferError
    ├── EOFError
    ├── ExceptionGroup [BaseExceptionGroup]
    ├── ImportError
    │   └── ModuleNotFoundError
    ├── LookupError
    │   ├── IndexError
    │   └── KeyError
    ├── MemoryError
    ├── NameError
    │   └── UnboundLocalError
    ├── OSError
    │   ├── BlockingIOError
    │   ├── ChildProcessError
    │   ├── ConnectionError
    │   │   ├── BrokenPipeError
    │   │   ├── ConnectionAbortedError
    │   │   ├── ConnectionRefusedError
    │   │   └── ConnectionResetError
    │   ├── FileExistsError
    │   ├── FileNotFoundError
    │   ├── InterruptedError
    │   ├── IsADirectoryError
    │   ├── NotADirectoryError
    │   ├── PermissionError
    │   ├── ProcessLookupError
    │   └── TimeoutError
    ├── ReferenceError
    ├── RuntimeError
    │   ├── NotImplementedError
    │   └── RecursionError
    ├── StopAsyncIteration
    ├── StopIteration
    ├── SyntaxError
    │   ├── IndentationError
    │   └── TabError
    ├── SystemError
    ├── TypeError
    ├── ValueError
    │   └── UnicodeError
    │       ├── UnicodeDecodeError
    │       ├── UnicodeEncodeError
    │       └── UnicodeTranslateError
    └── Warning
        ├── BytesWarning
        ├── DeprecationWarning
        ├── EncodingWarning
        ├── FutureWarning
        ├── ImportWarning
        ├── PendingDeprecationWarning
        ├── ResourceWarning
        ├── RuntimeWarning
        ├── SyntaxWarning
        ├── UnicodeWarning
        └── UserWarning

```

raise Exception

```
try:
```

```
    ; ← həy komputu namam  
buzurum uckorvare
```

```
except ZeroDivisionError:
```

```
    exit:
```

```
finally:
```

```
    ; ← he buu  
uckorvare
```

*system
error e i e
remains*

o modam ceyrae

```
class MyException(Exception):
    pass
```

```
raise MyException("Exception text")
```

```
except MyException as E22:
    print(E22)
```

```
...
```