Bonus
Учимся учиться?

Почему студенты отчисляются?

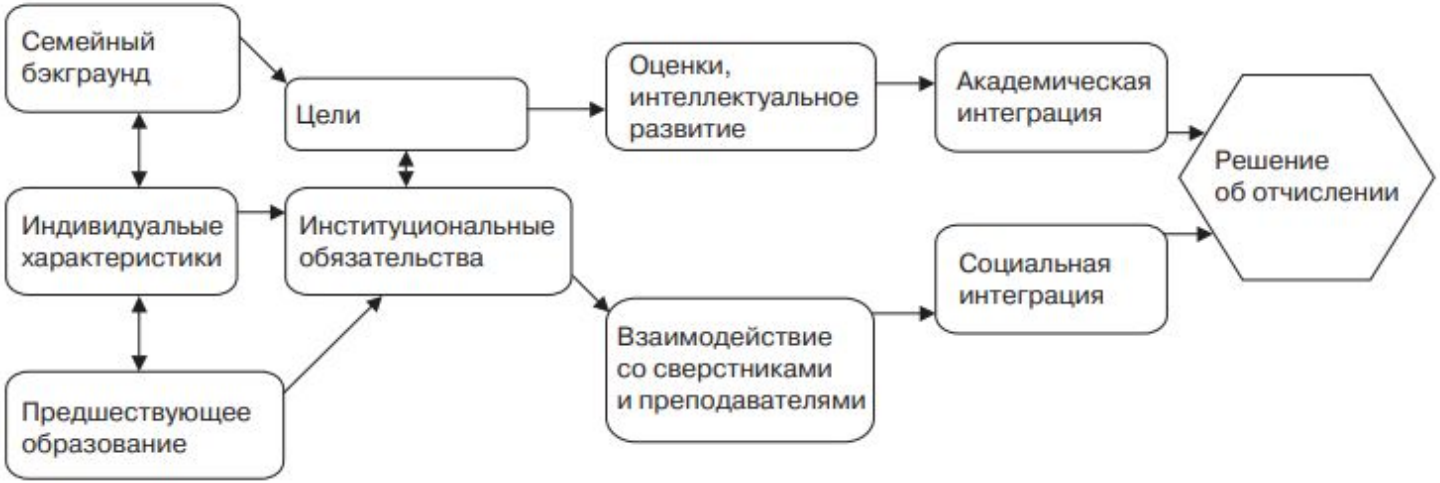
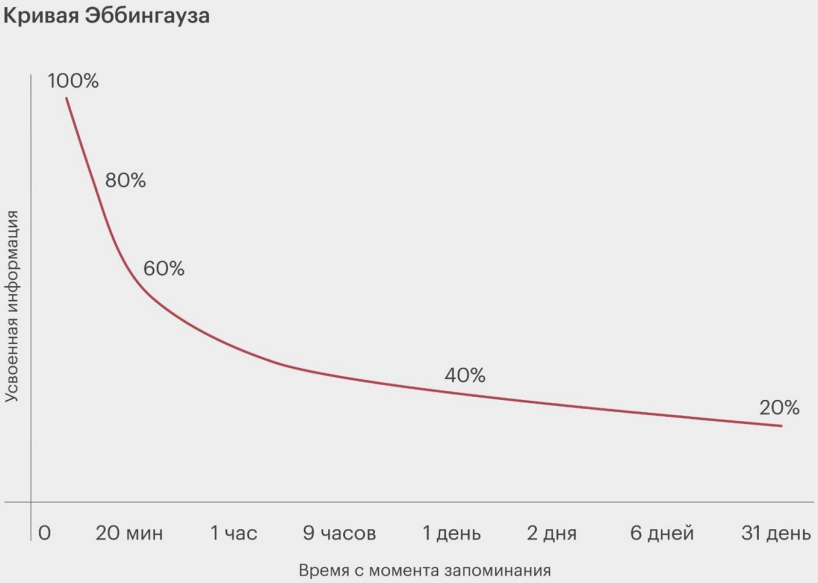


Рис. 1. Концептуальная схема отчисления из колледжа [Tinto, 1975, p. 95]

Треугольник “живучести”

Его автор Скот Свейл предлагает рассматривать следующие переменные: академическая подготовленность, атмосфера кампуса, социальная и академическая интеграция, финансовая помощь [Swail, 2004]. Эти факторы объединяются автором в три группы: познавательные, социальные, институциональные. Так получаются три стороны треугольника, обозначающего студенческий опыт в целом.

Кривые забывания и обучения



Модель памяти Аткинсона-Шиффрина

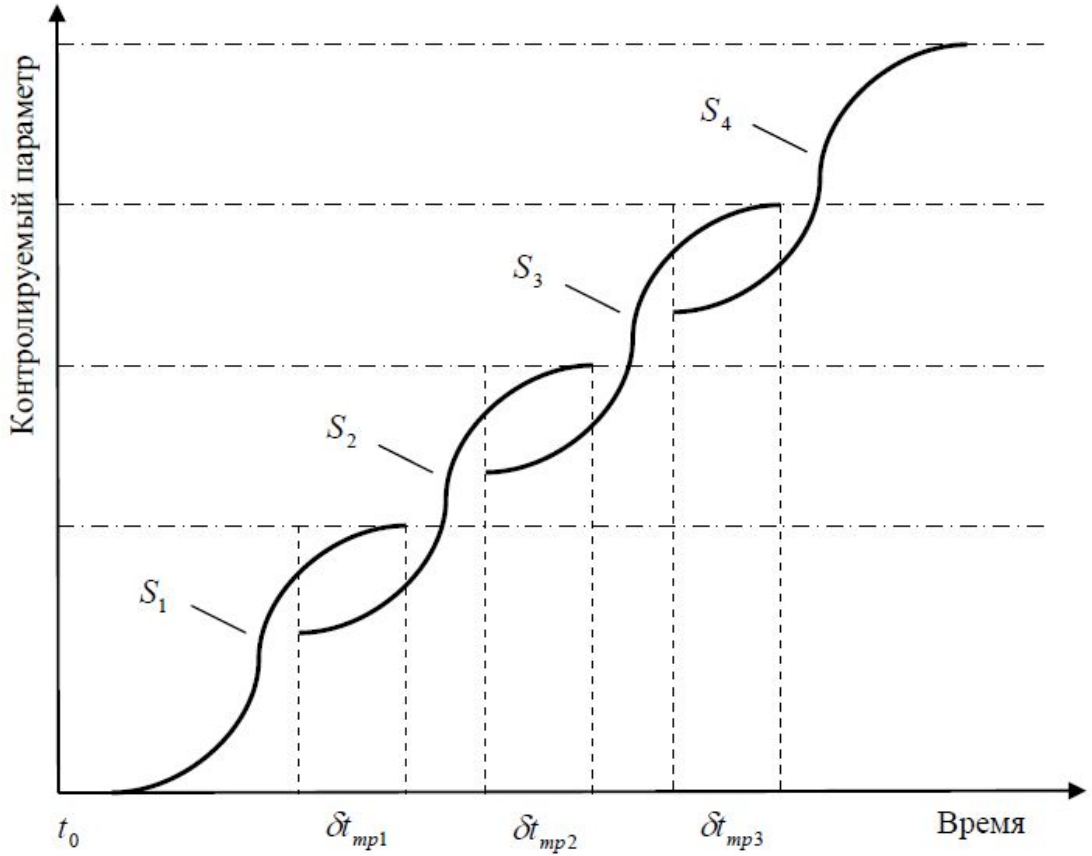
Чтобы хорошо запомнить

первое повторение — сразу по окончании чтения;

второе повторение — через 20—30 минут после первого повторения;

третье повторение — через 1 день после второго;

четвёртое повторение — через 2—3 недели после третьего;

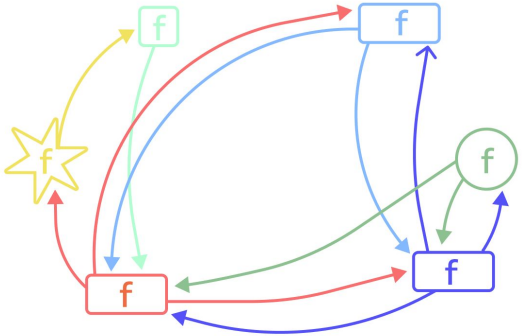

Контролируемый параметр
S_1
S_2
S_3
S_4
t_0
$\delta t_{тр1}$
$\delta t_{тр2}$
$\delta t_{тр3}$
Время

Лекция 9
Объектно-ориентированное программирование. Наследование

Еще раз про классы

Класс!

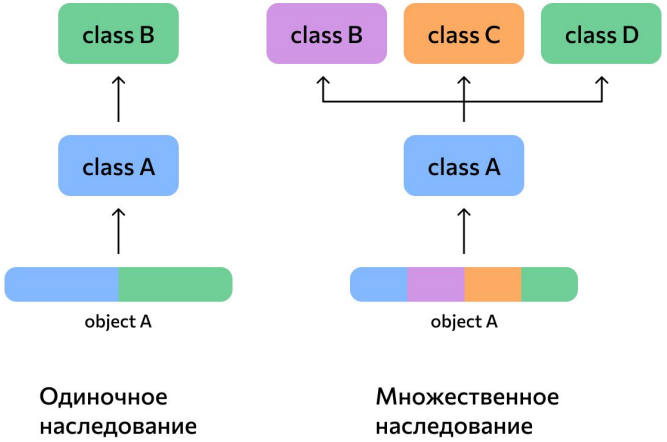
Зачем нужны классы? Классы – это способ единым, заданным программистом способом, взаимодействовать с типовыми унифицированными объектами. Это позволяет избежать, например, ситуаций, когда у вас набирается некоторый каскад функций, которых необходимо изменять под каждую конкретную задачу. Это может привести к такой структуре кода:



Одни функции вызывают другие и уже сложно понять, чем мы занимаемся.

Наследование

Наследование – способ доопределить класс,
добавив ему какие-то свойства. При конструировании объекта в объекте дочернего класса сначала конструируется та часть объекта, которая является представителем родительского класса, а потом “надстройка”, которую мы доопределили в дочернем классе



Дочерние классы наследуют свойства родительского

Множественное наследование

Мы можем унаследовать класс от нескольких родителей. В таком случае в качестве “фундамента” в объекте дочернего класса будут лежать все представители родительских классов.

```
class Employee:

    def __init__(self, name):
        self.__name = name

    @property
    def name(self):
        return self.__name

    def work(self):
        print(f"{self.name} works")


class Student:

    def __init__(self, name):
        self.__name = name

    @property
    def name(self):
        return self.__name

    def study(self):
        print(f"{self.name} studies")


class WorkingStudent(Employee, Student):
    pass


tom = WorkingStudent("Tom")
tom.work()      # Tom works
tom.study()     # Tom studies
```

Полезные ссылки

https://academy.yandex.ru/handbook/python/article/volshebnye-metody-pereopredelenie-metodov-nasledovanie