Даты

В этом листке вам нужно будет решать задачи, связанные с датами и временем, поэтому прежде всего нужно понять, как хранить даты и время в программе. Конечно, лучше всего для типа данных “Дата” создать класс.

class Date:

У объектов этого класса будет три поля: day, month, year.

Подумайте, какие операции можно выполнять с датами. Например, можно преобразовывать даты в строки для удобного вывода или, наоборот, преобразовывать строки в даты. К датам можно прибавлять (вычитать) целые числа (например, прибавить число 1 — это получить следующую дату). Наконец, можно брать разность двух дат (это количество дней между ними).

Для преобразования номера месяца в строку лучше всего использовать список строк-названий:

MonthNames = ('', 'January', 'February', 'March', 'April', 'May', 'June', 'July', 'August', 'September', 'October', 'November', 'December')

Для вывода названия дней недели также лучше использовать список:

DayOfTheWeek = ('Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday')

Данные списки лучше всего сделать глобальными и объявить в самом начале программы.

А вот для преобразования строки-названия месяца в число лучше использовать словарь:

MonthNums = {'January': 1, 'February': 2, 'March': 3, ...}

Инкапсуляция

Если имя поля начинается с двух символов подчеркивания, например, __day, __month, __year, то эти поля являются “скрытыми”, то есть доступ к ним можно получить только из методов класса. Для работы с этими полями нужно использовать методы класса: конструкторы, метод __str__, перегруженные операции и т.д. При необходимости явно получить или изменить значение поля пишутся специальные методы типа get_day или set_day.

Слоты классов

Если специальному атрибуту класса с именем __slots__ присвоить значение, в виде списка из нескольких текстовых строк, то только перечисленные в данном списке имена могут использоваться в качестве имен полей (атрибутов) классов. Например:

```
class Date:
  __slots__ = ['__day', '__month', '__year']
```

Это также приведет к уменьшению времени обращения к атрибутам класса за счет другого способа хранения атрибутов в экземпляре класса.