**Ежедневник / планировщик**

# Структуры данных для года

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Тип | Плюсы | Минусы | Комментарии |
|  | Массив из 365/6 элементов | * Фиксированный объём * Простота * Быстрый доступ к любой дате * Легко добавлять новую запись – просто идёшь по индексу | * Много места расходуется впустую – много ли??? * При генерации дат в годах до или после – надо генерировать целые массивы * Доп. обработка при чтении из файла, записи в файл 🡺 тратится время на поиск следующей активной записи * нужна функция перевода даты в индекс * обратный перевод индекса в дату сложнее | * Скорее всего, не так уж и много лишнего места расходуется. * Если делать день – структура, тогда при инициализации массива будет выделена память. * Если использовать классы, то память будет выделена при явном создании элемента * Всё равно же не будет прямого обращения к памяти, чтобы извлечь сколько-то дат подряд, соседнее расположение всех дней в физ. памяти не так критично |
|  | Зубчатый массив [12][28/29/30/31] | * Дата точно соответствует индексу =  [номер месяца] [день месяца] * Легко упорядочивать по датам внутри года и внутри месяца * Не нужна функция соответствия даты и индекса * Легко искать, т.к. быстрый доступ | * Возможно сложности при прокрутке – переход от месяца к месяцу ??? * Доп. обработка при чтении из файла, записи в файл 🡺 тратится время на поиск следующей активной записи | * Т.к. сохранение/чтение из файла не самая частая операция, то можно и потратить время * Всё равно же не будет прямого обращения к памяти, чтобы извлечь сколько-то дат подряд, соседнее расположение всех дней в физ. памяти не так критично |
|  | Коллекция из предустановленных  Это динамический массив List | * Легко загружать/выгружать из/в файл – нету пустых мест * Лёгкий переход к соседней записи, которая не обязательно в соседнем дне или месяце * Легко искать по содержанию записи – не надо проходить пустые места | * Долго искать нужный элемент по дате * Долго упорядочивать по датам |  |
|  | Собственный динамический список | * Same for collection, только ещё и самому надо будет всё реализовывать | * Same for collection, только ещё и самому надо будет всё реализовывать | * 🡪 зачем самому мучится, когда можно использовать существующие библиотеки |
|  | Годы – коллекция  Год – массив | * Между годами прыгать надо не так часто, поэтому время ползанья по списку не много * Не надо генерировать промежуточные годы, если даты разделены годами * Объединяем достоинства обоих подходов |  |  |
|  | Годы – двунаправленный динамический список Год - массив | * Т.к. лет немного, то относительно быстро будет переключение между годами. В основном это надо будет на рубеже двух годов | * Надо самому делать. Пока не нашёл предустановленных методов |  |

1. Day[] year = new Day[365/366] доступ - year[daynumber] daynumber == DateToNumber(dd.mm) == dayspassed[month] + day   
    dayspassed[] = {31, 59, 90, 120, 151, 181, 212, 243, 273, 304, 334} – 11 elements
2. Day[,] year = new Day[12,31]; доступ - year[month, day] надо контролировать месяцы по длине
3. Day[][] year = new Day[12][]; при появлении нового месяца инициализируем new Day[month][количество дней в месяце]   
    доступ - year[month][day] не надо вычислять отдельно индекс для дня и контролировать длину месяца
4. List<Day> year = new List<Day>(365) – так делать смысла нет, потому что это эквивалентно 1.   
   При выделении сразу 365/366 дней мы не будем прибавлять или убавлять дни.  
     
   Имеет смысл только в случае new List<Day>(), capacity 365, тогда можно добавлять или удалять дни.   
    Но тогда увеличивается время: поиска дня по дате, вставка нового дня, удаление старого дня.   
    При прокрутке по датам на экране тоже нет выигрыша – надо всегда проверять – дата на экране есть в списке или нет  
    Единственный выигрыш – сохранение в файл, чтение из файла, но он на столько незначителен …
5. List<Month> year = new List<Month>(12). year[m].month[d].day[recordNum]
   1. List<Day> month = new List<Day>(28/29/30/31)
   2. List<Record> day = new List<Record>(5)
6. Ежедневник - собственный динамический список из лет. Год – собственный динамический список из дней
7. List<Year> Dairy = new List<Year>(); class Year { массив одного из типов для года}

# Структура данных для месяца

Нет смысла писать структура данных для месяца

# Структура данных для дня

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | Тип | Плюсы | Минусы | Комментарии |
|  | Массив из 24 элементов - часов | * Фиксированный объём * Быстрый доступ к любому элементу * Легко добавлять новую запись * Уже упорядочены по времени * Легко искать, т.к. быстрый доступ | * Много места расходуется впустую – много ли??? * Доп. обработка «пустых (в которых нет событий) дат при чтении/записи из/в файл | * Если делать день – структура, тогда при инициализации массива будет выделена память. * Если использовать классы, то память будет выделена при явном создании элемента |
|  | Массив из 24 \*4 по 15 минут  \*6 по 10 минут |  | * Увеличивается время на 1 день |  |
|  | List <Items> из предустановленных | * Легко загружать/выгружать из/в файл – нету пустых мест | * Долго искать нужный элемент * Долго упорядочивать по датам |  |

# Структура данных для одной записи

Что такое одна запись?

|  |  |  |
| --- | --- | --- |
| **My** | **Google** | **Microsoft** |
| Event (Business event, Meeting, Appointment)– привязка ко времени (когда произойдёт)  Note – просто текст (привязка ко времени по дате и времени создания)  To do action – просто список (привязка ко времени, когда надо начать/закончить) | Event / Мероприятие | Задачи/ Task |
| Любая запись – Stack – для отката изменений |  |  |

* Привязанная ко времени (часам и минутам): Событие, дело, встреча (appointment)
* Не привязанная ко времени – в течение дня либо просто заметка
* Это просто любая заметка?
* Это дело, которое надо сделать в определённое время
* Это appointment
* Дело и appointment - это

Функционал

В ежедневнике реализовать возможность

- создания

- удаления

- редактирования

записей

В отдельной записи должно быть не менее пяти полей

Реализовать возможность

- Загрузки данных из файла

- Выгрузки данных в файл

- Добавления данных в текущий ежедневник из выбранного файла

- Импорт записей по выбранному диапазону дат

- Упорядочивания записей ежедневника по выбранному полю