Задача 5_2 Генератор компараторов

Давайте представим, что вы разрабатываете инновационный сервис поиска авиабилетов AviaScanner. В вашем сервисе авиабилет представляется в виде структуры

```
1  struct Date {
2  int year, month, day;
3  };
4
5  struct Time {
6  int hours, minutes;
7  };
8
9  struct AirlineTicket {
10 string from;
11 string to;
12 string airline;
13 Date departure_date;
14 Time departure_time;
15 Date arrival_date;
16 Time arrival_time;
17 uint64_t price;
18 };
```

В данный момент вы работаете над функцией сортировки результатов поиска. Пользователь вводит свой запрос и получает список подходящих билетов. Дальше он может задавать параметры сортировки этого списка. Например, сначала по цене, затем по времени вылета и, наконец, по аэропорту прилёта.

Чтобы реализовать сортировку как в примере, можно воспользоваться алгоритмом цифровой сортировки

```
1  void SortTickets(vector<AirlineTicket>& tixs) {
2  stable_sort(begin(tixs), end(tixs), [](const AirlineTicket& lhs, const Air-
   lineTicket& rhs) {
3  return lhs.to < rhs.to;
4  });
5  stable_sort(begin(tixs), end(tixs), [](const AirlineTicket& lhs, const Air-
   lineTicket& rhs) {
6  return lhs.departure_time < rhs.departure_time;
7  });
8  stable_sort(begin(tixs), end(tixs), [](const AirlineTicket& lhs, const Air-
   lineTicket& rhs) {
9  return lhs.price < rhs.price;
10 });
11 }
```

Как видите, в примере выше есть дублирование кода — нам пришлось написать три лямбда-функции, которые отличаются только полем, по которому выполняется сортировка. От этого дублирования можно избавиться, написав макрос SORT_BY и применив его следующим образом:

```
1  void SortTickets(vector<AirlineTicket>& tixs) {
2  stable_sort(begin(tixs), end(tixs), SORT_BY(to));
3  stable_sort(begin(tixs), end(tixs), SORT_BY(departure_time));
4  stable_sort(begin(tixs), end(tixs), SORT_BY(price));
5  }
```

Напишите макрос SORT_BY, который принимает в качестве параметра имя поля структуры AirlineTicket. Вызов sort(begin(tixs), end(tixs), SORT_BY(some_field)) должен приводить к сортировке вектора tixs по полю some_field.

Вам дан файл airline_ticket.h, содержащий объявления структур Time, Date и AirlineTicket, а также заготовка решения в виде cpp-файла sort_by.cpp. Пришлите на проверку cpp-файл, который

- подключает заголовочный файл airline_ticket.h
- содержит макрос SORT_BY
- содержит определения операторов, необходимых для использования классов Date и Time в алгоритме сортировки и макросе ASSERT_EQUAL (формат вывода в поток можете выбрать произвольный)

airline_ticket.h

sort_by.cpp