Consider the following struct:

```c
struct date {
    int year;
    char *month;
    int day;
};
```

1. Write a function with the signature
   ```c
   void createDate(struct date **d, int year, char* month, int day)
   ```
   which allocates space for a new `struct date`, stores the pointer to it in `*d`, and initializes the fields of the new struct with the parameters passed to `createDate`. You must not allocate more space than you need. Note that you cannot just assign `char* month` to the `month` field.

2. Write a function with the signature
   ```c
   void getDate(struct date **d)
   ```
   which reads a new date from the keyboard (you decide how), allocates space for the new date, and stores a pointer to it in `*d`. You must not allocate more space than you need.

3. Write a function with the signature
   ```c
   void copyDate(struct date **dest, struct date src)
   ```
   which copies the date `src` to `*dest`. You should allocate `*dest` before copying `src` to it. Make sure you copy the month correctly.

4. Write a function that would make freeing the allocated memory blocks allocated by the functions from questions 1 to 3 easy.

5. Write a function that increments a date by one day

6. Write a function with the signature
   ```c
   int compareDates(struct date *d1, struct date *d2)
   ```
which returns -1 if d1 comes before d2, 1 if d1 comes after d2, and 0 if d1 and d2 are the same. You can assume all the months are written in all lowercase letters (e.g., “january”, “february”, and so on)