Computer Architecture
Homework no 5
Linked lists of character strings in C

Submission due: Tuesday 21 March

We work with the following data type, which describes the type of a linked list of character strings:

```
struct cell{
  char *key;
  struct cell *next;
};
```

Exercise 1.

§1-1. Write a function

```
struct cell *cons(char *string, struct cell *list);
```

which appends a copy of the string `string` to the beginning of the list `list` and returns the result. You are allowed to use the function `strdup` to copy a string.

§1-2. Write a function

```
void print_list(struct cell *list)
```

which prints the elements of a list, one per line.

§1-3. Write a function

```
int list_length(struct cell *list)
```

which returns the length of the list given as parameter.

Exercise 2.

§2-1 Write a function `main` and test the code which you have just written by constructing the linked list of three character strings:

```
["Beautiful","Weather","Today"]
```
(1)

§2-2. Write a function

```
free_list(struct cell *list)
```

which frees the memory allocated for a list of character strings.

§2-3. Modify your `main` function so that it frees all the memory allocated in memory for the linked list of character strings (1) constructed above in §2-1.

Exercise 3.

§3-1. Write a function

```
int list_member(char *string, struct cell *list)
```

which returns true if the string `string` is an element of the list `list` without taking care of the difference between the letter case. (You can use the function `strcasecmp` which you will find in the library `strings.h` for that purpose).

§3-2. Check that the character string `"Weather"` is an element of the linked list

```
["Beautiful","Weather","Today"]
```

constructed in §2-1 while the character string `"Cloudy"` is not an element of the list.

Exercise 4. [optional]

§4-1. Write a function

```
struct cell *read_words(char *filename)
```

which opens the file `filename` in read mode, and stores every line which it contains in a linked list of character strings. Do not forget to close the file in the end.

§4-2. Write a function `main` which reads the file

```
/usr/share/dict/words
```

and stores the content in a linked list, and then prints the length of it.

§4-3. Write a spellchecker which reads the file `/usr/share/dict/words` and stores its content in a linked list. The spellchecker will then read words from the standard input and for each of them, indicate to the user whether the word is in the dictionary or not.

§4-4. Modify your program so that it can input a full text (with several words per line and punctuation marks) and then print the list of words which are not in the dictionary.