To apply to Scalien, please solve these two problems.

Once you're done, put the solutions into two folders, put your CV (pdf) next to it, zip it up, and send it to jobs@scalien.com. We will get back to you. Please keep in mind:

- write the programs in english
- test the programs before submitting
- supply your test-cases
- supply Makefiles
- you do not have to supply any documentation, just put some comments where appropriate
- we recommend you wait two days after completing the assignment before submitting – you may discover some bug(s) while thinking about it in the background
- don't forget your CV (pdf)!

1. Remove character

Write a C routine with the following prototype:

```
void remove_char(char *str, char c);
```

The routine modifies the given zero-terminated string in place, removing all instances of the given character.
For example:

```
char* str = (char*) malloc(128);
strcpy(str, "zzzhelzzzlozz");
remove_char(str, 'z');
printf(str); // prints "hello"
```

2. Intrusive stack

Write a templated intrusive stack class `InStack` in C++. An intrusive container is one where the programmer puts the necessary pointers into the contained classes. For example, if the programmers want to store `Points` in the intrusive stack, he has to put:

```
class Point
{
	double x, y, z;
	// etc ...

	// so a Point can be linked into an intrusive stack:
	Point *next;
};
```

Intrusive containers avoid an extra `malloc()`, because no internal "node" structure has to be allocated (or declared). The user allocates the contained class (eg. a `Point`), passes it to the stack, and the stack just chains it by setting the appropriate pointers.

The class should support mutations:

- `void Push(T*)`
- `T* Pop()`
- `void Clear()`

Iteration:

- `T* Top()`
- `T* Next(T*)`

Retrieve number of items:

- `unsigned GetLength()`