Questions week 5

1. Assume you have the following macro:

```c
#define square(a) ((a) * (a))
```

Why is it a bad idea to use it for each of the following arguments?
- `square(++x)`
- `square(cos(x))`

*page 280, 403*

2. Although it is uncommon, suppose you wish to put multiple statements in an object-like macro. How does the idiom `do-while (0)` help?

*page 405*

3. What is the difference between the following two `include`-directives?

```c
#include "a.h"
#include <b.h>
```

In the second form, how can you tell the compiler where to look for the file `b.h`?

*page 409*

4. Is `inline` a directive the compiler must follow or only a hint to the compiler?

*page 334*

5. Why are inline functions with external linkage not allowed to declare modifiable data with static storage duration?

*page 334*

6. Why are inline functions with external linkage not allowed to access any symbol with internal linkage (i.e. either a function or variable declared with `static`)?

*page 334*

7. Why do the above restrictions not apply to inline functions with internal linkage?

*page 334*

8. What is the purpose of the macro `assert` and how can you disable it to improve the speed of the program?

*page 418*
9. Which floating point exceptions are recognized by C (and all other languages which support IEC 559 such as C++ but notably not Java). In an exception?

   page 432

10. How can you detect that your program triggered a floating point exception?

    page 433-434