Questions week 3

1. Which two main approaches to use non-Latin characters, such as å, ä, and ö in strings are available in C?

   page 248

2. **Scopes of identifiers**. What does the following program print and why?

   ```
   #include <stdio.h>

   int main(void)
   {
           int     i = 8;

           for (int i = 0; i < 10; ++i)
                   ;

           printf("i = %d\n", i);

           return 0;
   }
   ```

   page 253

3. What does **linkage** mean and how can we specify that a function has internal linkage? Why can that be useful?

   page 254

4. Many programmers have ”discovered” (wrongly) that it’s possible to use a cast in order to let an integer pointer point to a floating point variable. For example:

   ```
   float           x;
   unsigned int*   p = (unsigned int*)&x;

   *p = 0x12345678;
   ```

   What is the rule called which makes such code meaningless?

   page 267

5. Storing and later printing a small negative value (such as -5) in an unsigned integer type results in a large value. Why is that?

   page 276

6. What does the following program print and why? What is this processing of the operands in an arithmetic operation called?

```
#include <stdio.h>

int main(void)
{
        unsigned char   a = 255;

        printf("%d\n", a - 256);

        return 0;
}
```

page 276

7. What is a **bit-field** in a struct and why is it usually important to be specific about the sign of an `int`, i.e. using either `signed int` or `unsigned int` for bit-fields?

page 322

8. Two ways of introducing integer constants are with `#define` and with `enum`. Which important restriction on the values do enums impose?

page 328