These exercises are mostly theoretical in nature. Some of them requires you to do some research on your own, since not everything is covered during the seminars.

1. Given the following function overloads:

```
void fun(double, int, int);     // #1
void fun(int, int, double);     // #2
void fun(double, int, double); // #3
```

   For these function calls, determine which – if any – of the overloads above is called:

```
fun(0.0, 0.0, 0.0); // a
fun(0, 0, 0);       // b
fun(0, 0.0, 0.0f);  // c
fun(0.0, 0.0, 0);   // d
fun(0.0, 0, 0.0f);  // e
```

   Explain for each of them *why* the overload is called or why it failed.

2. Explain all type conversions that occur in this example.

```
#include <iostream>
#include <string>

int sum(double const* numbers,
unsigned long long size)
{
  double result{};
  for (unsigned i{}; i < size; ++i)
    result += static_cast<int>(numbers[i]);
  return result;
}

int main()
{
  std::string message{};
  message = "Enter a number: ";

  double numbers[3];
  for (int i{0}; i < 3; ++i)
  {
    std::cout << message;
    if (!(std::cin >> numbers[i]))
      return true;
  }

  std::cout << sum(numbers, 3) + 1.0 << std::endl;
}
```

3. Why do you think it is well-defined behaviour what happens when an unsigned integer value under- or overflows?

   Why is the same not true for signed integer values?

   I.e. why is the C++ standard comfortable with defining what happens if you subtract one from the smallest, or add one to the largest, possible unsigned value but *not* for signed?

4. Study the code below:

```
1 #include <cstdlib> // for std::abort()
2 #include <iostream>
3
4 int main()
5 {
6     std::cout << "Hello ";
7     std::abort();
8     std::cout << "there!" << std::endl;
9 }
```

   (a) Explain why the code (likely) **DOESN'T** print `Hello` to the terminal, even though the program isn't aborted until *after* the output statement.

   (b) How do you fix this program (without removing any code and without changing the intended behaviour of the program) so that it prints `Hello` before it crashes?

   Hint: There are at least *two* fundamentally different ways of fixing this.