Solutions to sheet 1::

Question 1:

The program below is supposed to print “Yes” if the two inputs are equal, “No” otherwise.

```
#include <iostream>
using namespace std;
int main()
{
int a,b;
cin >> a >> b;
if (a = b)
cout << "Yes" << endl;
else
cout << "No" << endl;
}
```

- **1. For what values of a and b will the program print ”Yes”?**
  As long as 'b != 0', the program will keep printing 'yes'.

- **2. Does the program work as mentioned? If not, what is the bug?**
  Nein, the program does not work as intended, since the '=' sign is a assignment operator and not a comparative operator.

Question 2:

2.1 How is a reference variable different from a pointer variable?

A reference is simple a alias for and existing variable. After a reference has been assigned or initialized to a variable, then it cannot be changed. \

A pointer however stores the memory address a variable and thus 'points' to a variable instead. A pointer can however be changed later.

Thus, a reference is similar to a 'const pointer'.

2.2 C++ favors the use of reference to pointer. Can a reference variable take a null value?

No, reference variables cannot take a NULL value. A pointer however point to a NULL value.

2.3 What is the difference between C-style strings and C++ style-strings ?

- In C, strings are represented by `char` arrays, and thus end with a NULL byte. AS such, herein memory management is the responsibilty of the coder. This can be done with a simple `malloc`.

- In Cpp:

-- strings `[std::string]` are objects and thus, memory management can be done in more easy and safe manner.

-- Cpp strings also support various mainupulation functions such as append, copy, concatenation, find etc.

2.4 What is a namespace? Why are they useful?

A namespace is a declarative region that provides a scope to the identifiers (the names of types, functions, variables, etc) inside it. Namespaces are used to organize code into logical groups and to prevent name collisions that can occur especially when your code base includes multiple libraries.

2.5 The program fragment

```
auto a; cin >> a;
```

throws a compilation error a is not initialized. How do you fix this error?

Here the variable a has been declared but not initialized. The error can be solved by simpy initialising it with some value, else it will pick up some random garbage value.

```
auto a; \\ variable is declared
a = 0;  \\ variable has been initialised
cin >> a;
```