

TDDE18 & 726G77

Programming in C++

# Course website

LIU ► IDA ► Undergraduate ► Courses ► TDDE18 ► Current Page In Swedish

**TDDE18 2017**

Syllabus  
Registration & Examination  
Activities & Rules  
Timetable & Deadlines  
Book Recommendations  
Slides  
Contact  
FAQ  
All Messages

**LAB SETUP**  
GNU GCC (Required)  
Start a new lab assignment (Required)  
Visual Studio Code (Recommended)

**LAB WORK**  
Lab signup  
Lab assignments  
Lab submission  
Assessment protocol  
Compilation and more  
Brief style guide  
Rules and Policy

**EXAM**  
Computer exam  
Allowed aids  
Previous exams

**INTERNAL**  
IDA internal

## TDDE18 Programming (C++) (6 ECTS)

Ht1-Ht2 2017

### Latest News...

28 / 8 2017	WebReg is open for lab registration Lab signup is now open for registration.
01 / 8 2017	Course start 2017 The first lecture take place Monday 28 / 8 15:15 in Ada Lovelace (Visionen) with course information, introduction to programming C++ in our computer environment.

Page responsible: Sam Le  
Last updated: 2017-08-06

All information you need to complete the course exists on the course website

# Administration

- Examiner – Klas Arvidsson
- Course leader – Sam Le
- Assistant 1 – Viktor Olsson
- Assistant 2 – Mladen Nikic
- Assistant 3 – Jonas Lind
- Assistant 4 – Fredrik Adolfsson

# Example from previous exam – The Quiz

## Introduction

Sam and his colleagues had a Christmas party quiz and he thought that it would be awesome to use the data from this quiz as an exam question.

## Functional requirements

Create a program that allows the user to enter the file name of the quiz data as a *command line argument*. The program must print out the top 3 participant with the highest total score.

```
$ ./a.out quiz_data.txt
```

1st place: August with 12160 total points

2nd place: Chris-Cross with 11944 total points

3rd place: Eric with 11623 total points

# Course layout

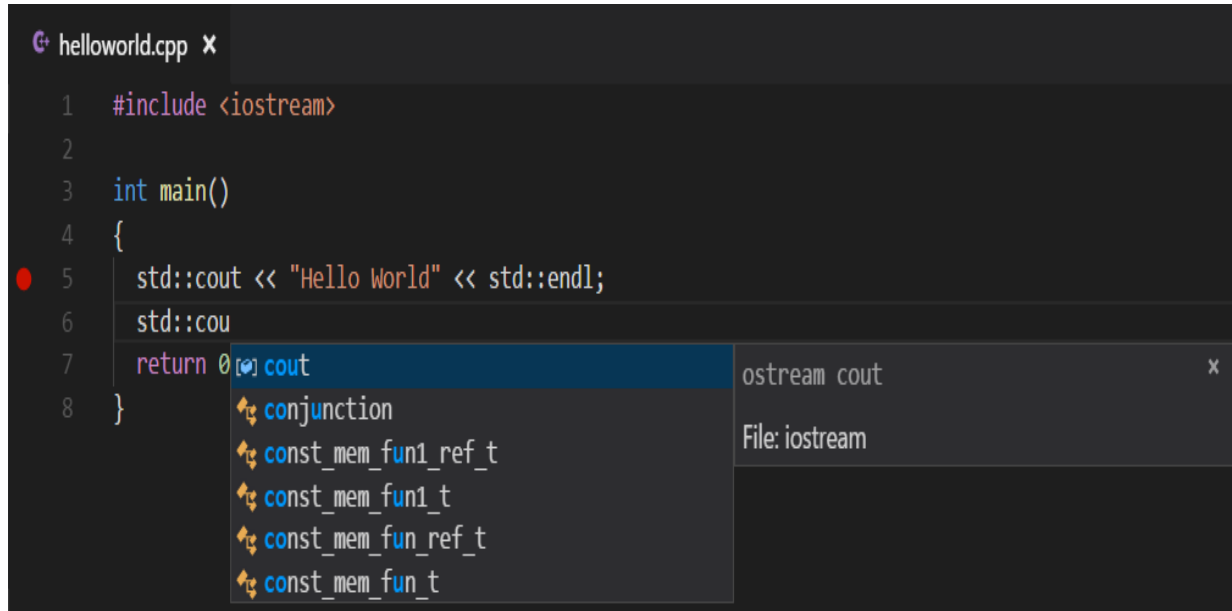
- Lectures
- Lessons
- Labs
  - 6 labs
- Exam

# Lab soft deadlines

- Bonus time to the exam for higher grade
- 5 extra minute per deadline
- 1 deadline per lab (1 – 6)
- 1 complementary work per lab
- You must demonstrate your work for the assistant.

# Visual Studio Code

```
helloworld.cpp x
1 #include <iostream>
2
3 int main()
4 {
5     std::cout << "Hello World" << std::endl;
6     std::cou
7     return 0;
8 }
```



- IntelliSense
- Debugging
- Built-in Git
- Extensions

<https://code.visualstudio.com/>

# Git + Sendlab

- Git – Used as a version control system and lab collaboration between you and your lab partner
  - [Try git](#)
- Sendlab – Used to submit your most recent code on git



# Sendlab

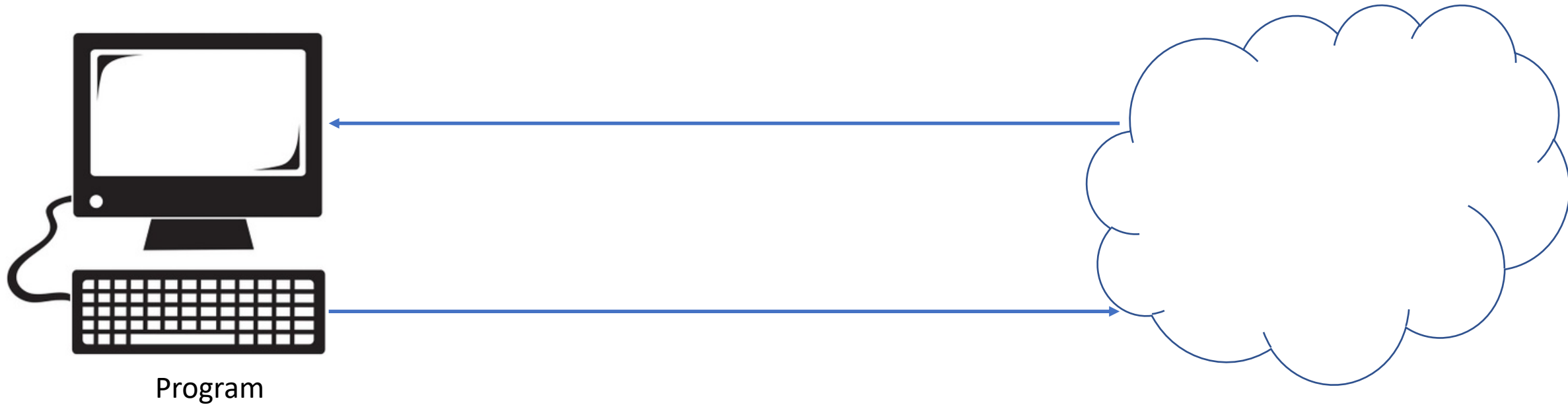
- `~TDDE18/sendlab registration` – registering for lab work
- `~TDDE18/sendlab start` – starting a lab
- `~TDDE18/sendlab send` – submitting a lab

main is the start button

```
int main() {  
}
```



# Input and output



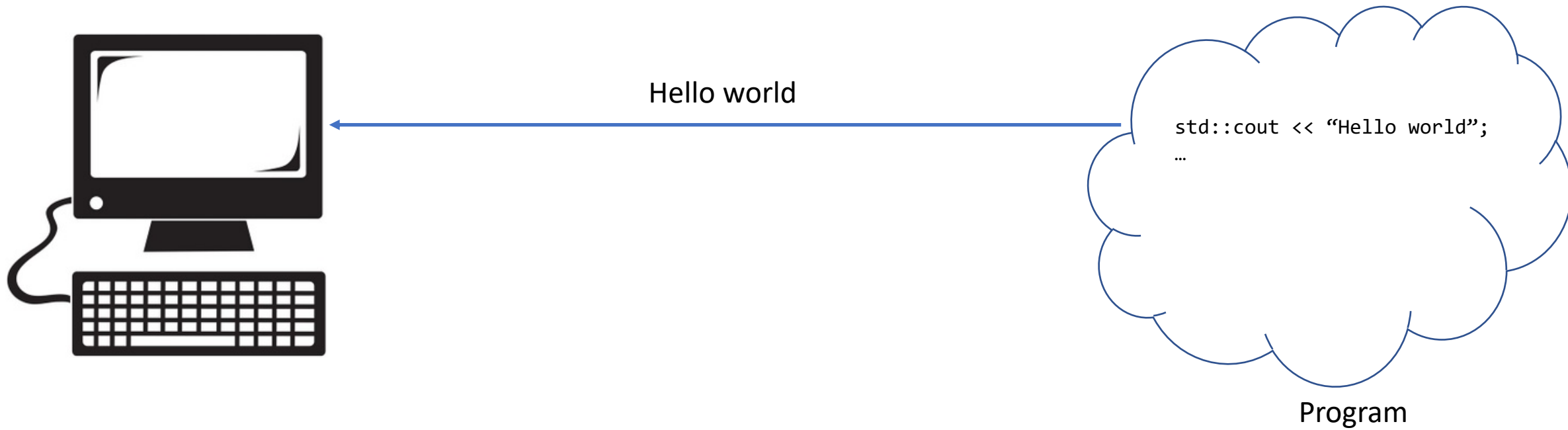
# Output buffer



# Cout

```
int main() {  
    std::cout << "Hello world";  
    ...  
}
```

# Output buffer



# Flush buffer

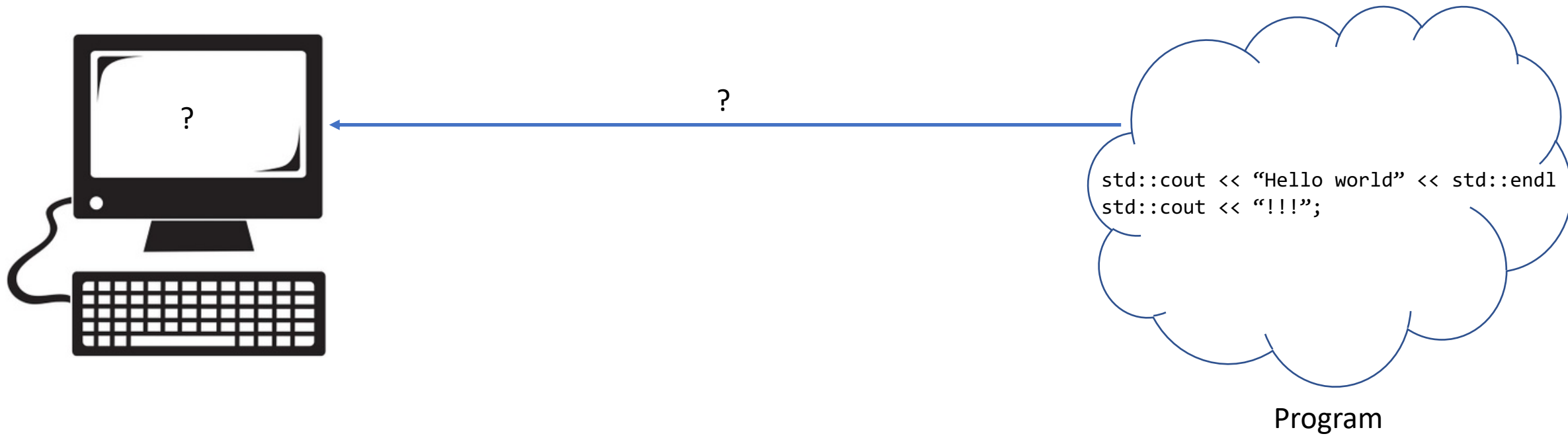
- When the program exits
- Use something to flush
  - endl
  - flush

# Flush buffer

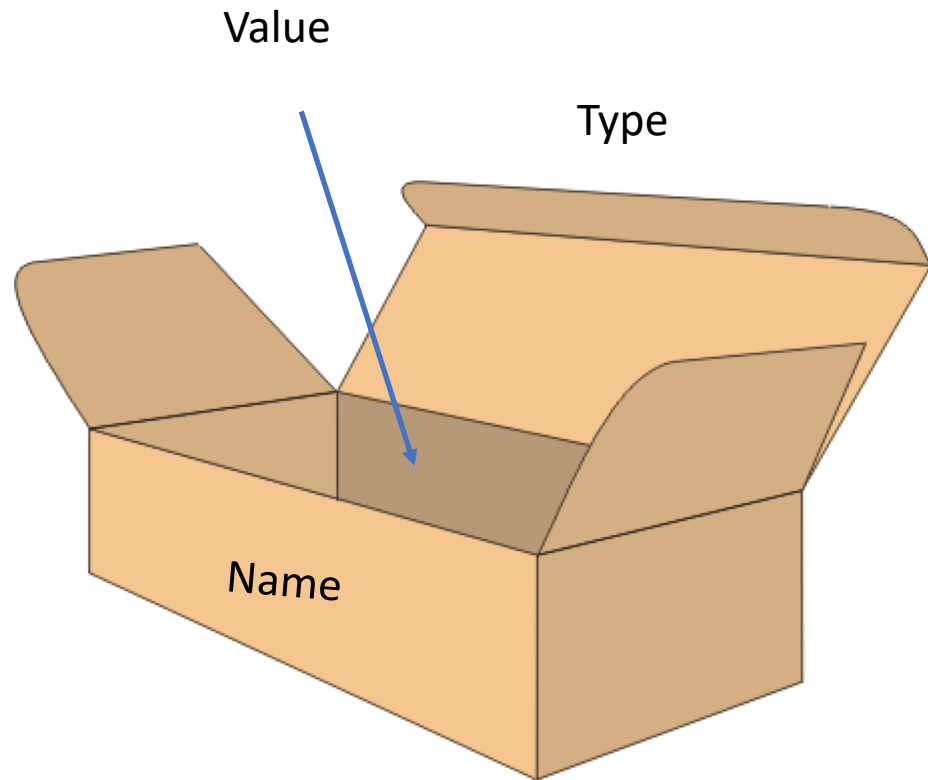




# Flush buffer



# Variables



Example:

- `int x{3}`
- `double y{3.14}`
- `char z{'s'}`

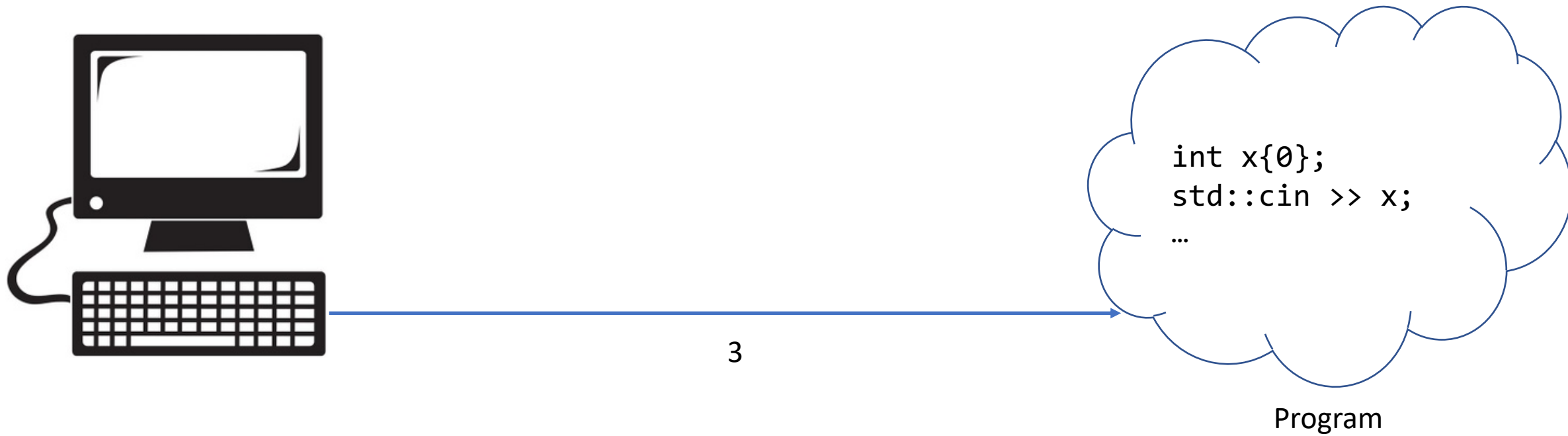
# Input buffer



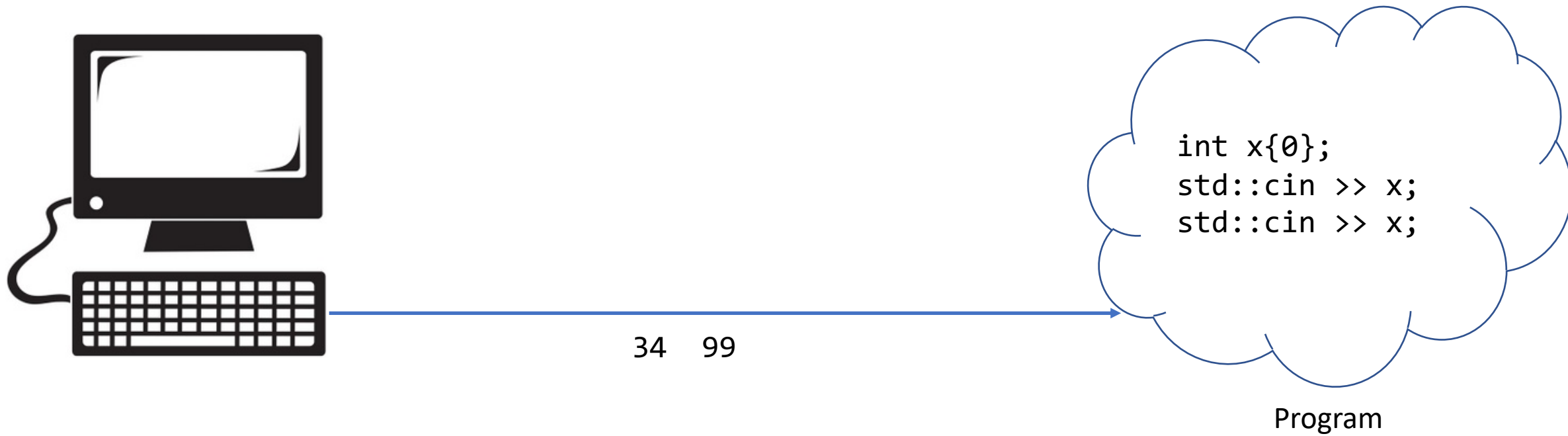
# Cin

```
int main() {  
    int x{};  
    cin >> x;  
    ...  
}
```

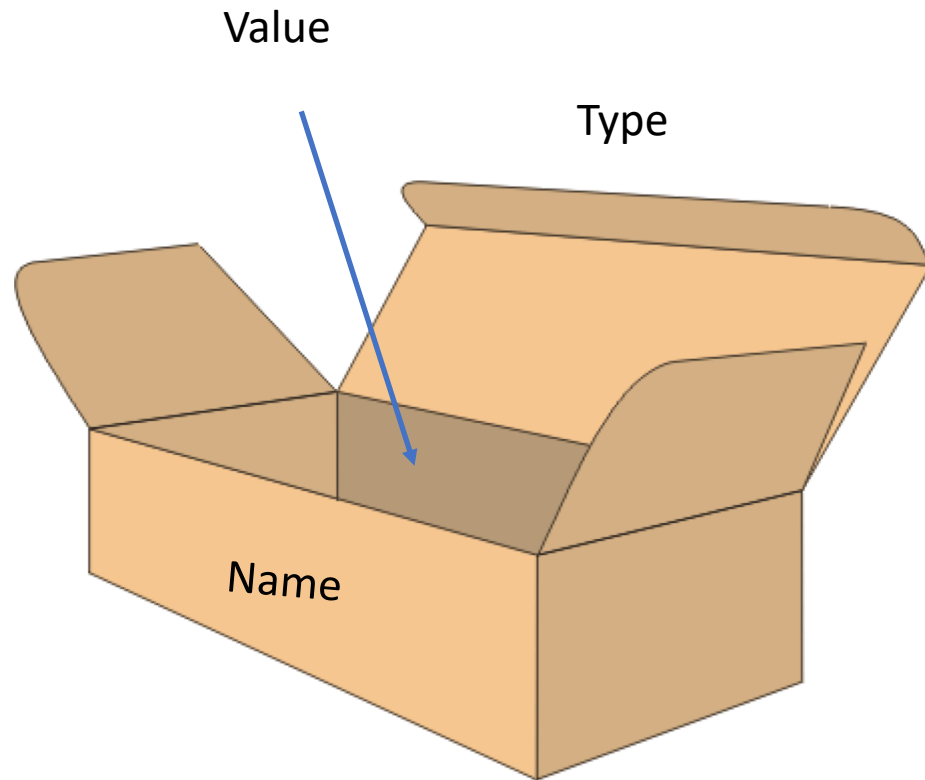
# Input buffer



# Input buffer



# String

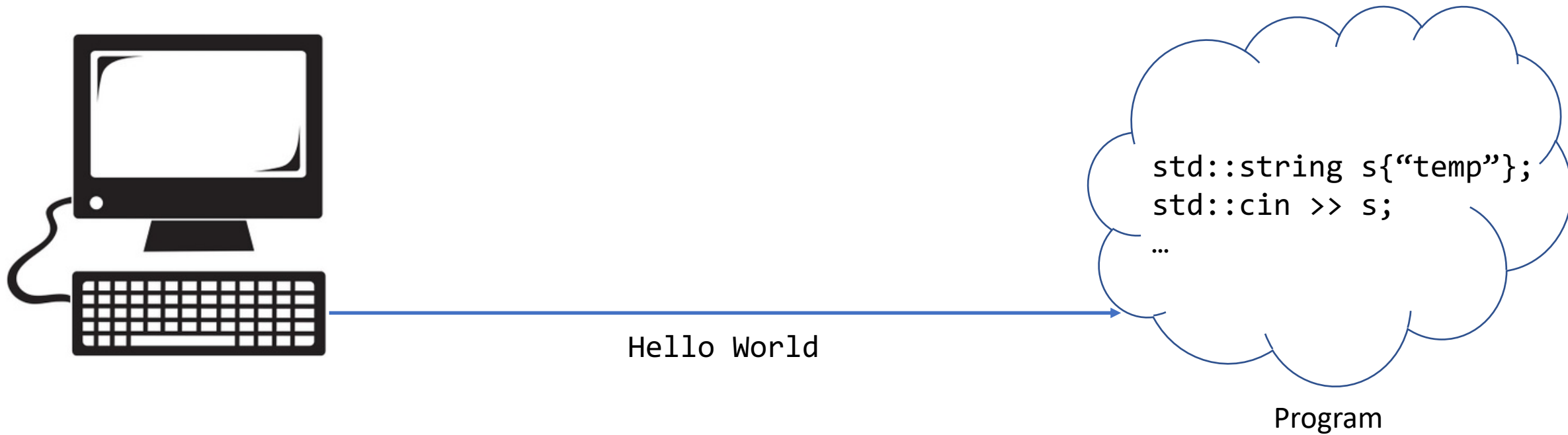


```
string s{"hello"}
```

```
s.size()
```

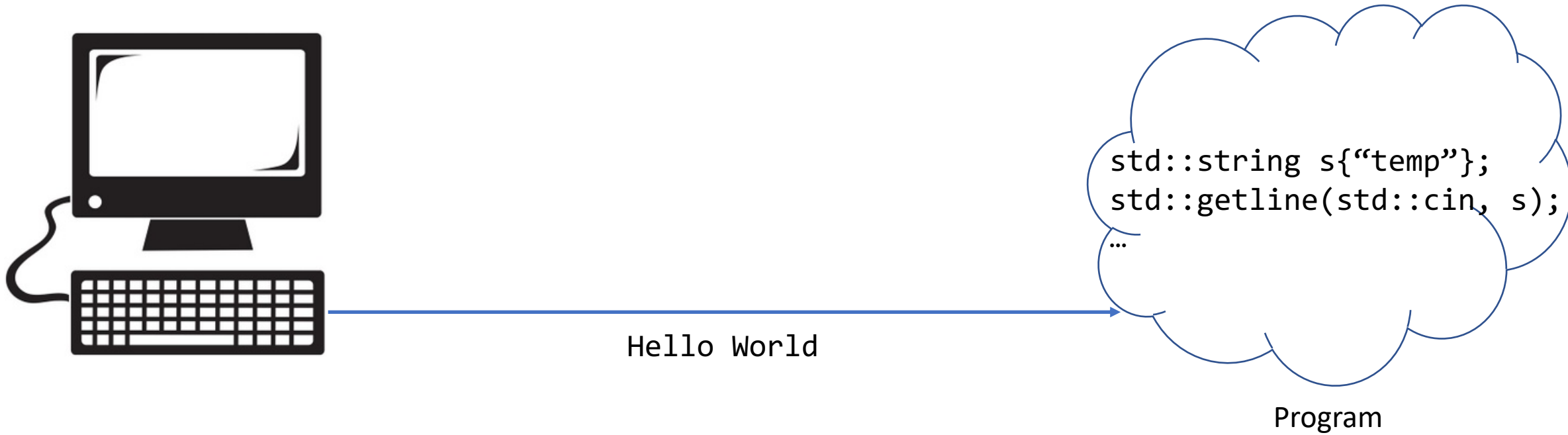
```
s.front()
```

# Input buffer





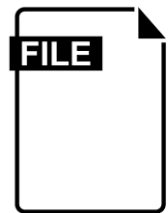
# Getline



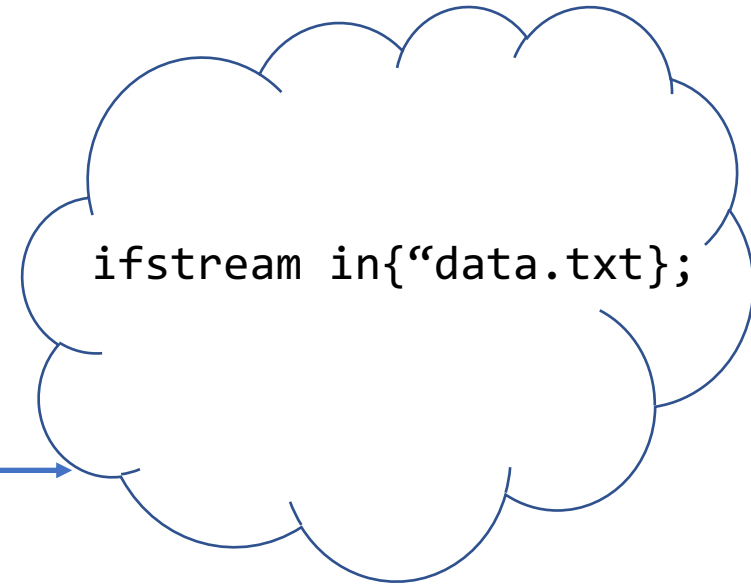
# Getline



# Reading from files

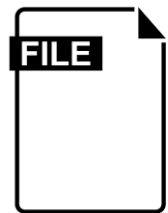


data.txt

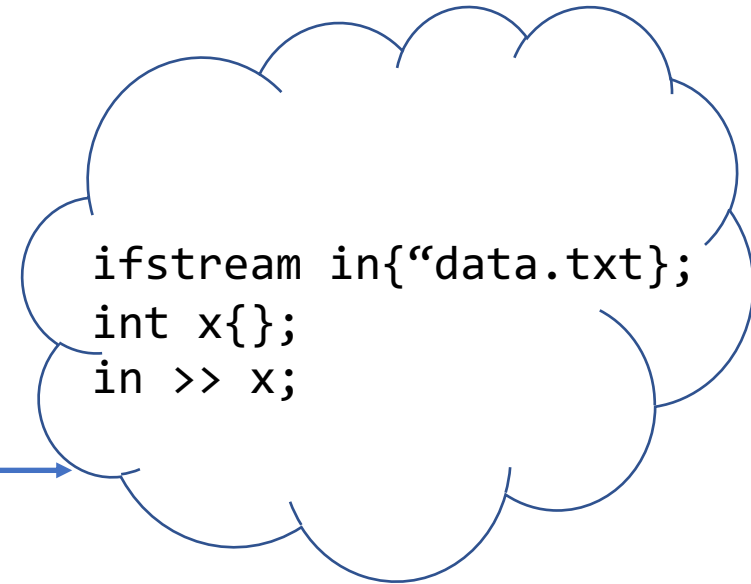


Program

# Reading from files



data.txt



Program

# Includes

- `iostream`
  - `cin`
  - `cout`
- `iomanip`
  - `setw`
  - `setfill`

```
#include <iostream>

int main() {
    std::cout >> "Hello world" >> std::endl;
}
```

# Namespace

```
#include <iostream>
using namespace std;

int main() {
    cout >> "Hello world" >> endl;
}
```

# Conditional statements: if/else if/else

```
if (some logical statement) {  
    do this  
}  
else if (some other logical statement) {  
    do this instead  
}  
else {  
    when all else fails do this  
}
```

# Comparison and Logical operators

- `a == b`
- `a != b`
- `a < b`
- `a <= b`
- `a > b`
- `a >= b`

`a = 1, b = 2`

- `a == b and c != b`
- `a == b or a == c`
- `!a`
- `&&` is equivalent to and
- `||` is equivalent to or

`c = 3, d = 4`



# Code example

```
int a{2};  
int b{2};  
if (a < b) {  
    cout << "This will not be executed" << endl;  
}
```

# Code example

```
int a{2};  
int b{2};  
if (a > b and a == b) {  
    cout << "This will not be executed" << endl;  
}
```

# loops

- for loops
  - while loops
  - do-while loops
- 
- Which one to use depends on purpose and readability

# For loops

- You know exactly how many times you want to loop

```
for (initializing; conditional statement; incrementing) {  
    body  
}
```

# Code example

```
for (int i{0}; i < 5; ++i) {  
    cout << i << " ";  
}
```

# While loops

- When you do not know how many times it will run

```
while (conditional statement) {  
    body  
}
```

# Code example

```
int a{};
```

```
cin >> a;
```

```
while (a < 10) {
```

```
    cin >> a;
```

```
}
```

# Do-While loop

- Run the body at least once

```
do {  
    body  
} while (conditional statement);
```



# Code example

```
do {  
    cout << "Enter a number between 0 and 10: ";  
    cin >> integer;  
} while (integer < 0 and integer > 10);
```

# Arithmetic operators

$+$ ,  $-$ ,  $*$ ,  $/$ ,  $\%$

Example:

- $1 + 3$
- $a - b$
- $c * d$
- $10.0 / 3$
- $3 \% 2$

# Arithmetic operators

`+=, -=, /=, *=`

`++, --`

Example:

`a += 4; => a = a + 4`

`b++ => b = b + 1;`

`--a => a = a - 1;`

```
int a{0};
```

```
int b{1};
```

```
int c{a++}; // What is c?
```

```
c = --b; // What is c?
```

# Type casting

Problem:

$$3 / 2 = 1$$

but

$$3 / 2.0 = 1.5$$

Example:

```
int a{3};
```

```
int b{2};
```

```
cout << a / b << endl;
```

**output: 1**

# Type casting

`static_cast<new type>(input)` // will return a value of the new type

Example:

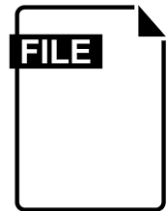
- `static_cast<int>('a');` // 65 due to ascii table
- `static_cast<double>(1);` // float value 1.0
  
- dont use c-cast eg: `(double)a`

# Commenting

// line comment

```
/*  
    multiline  
    comment  
*/
```

# Reading until end of input



data.txt



Program

# Reading until end of input

```
ifstream ifs{"data.txt"};
```

```
string s{};
```

```
while (...) {
```

```
    ...
```

```
}
```



# Reading until end of input

```
ifstream ifs{"data.txt"};
```

```
string s{};
```

```
while (/* As long that it can read from ifs to s*/) {
```

```
    ...
```

```
}
```

# Reading until end of input

```
ifstream ifs{"data.txt"};
```

```
string s{};
```

```
while (in >> s) {
```

```
    ...
```

```
}
```

# Lab 1

- Tuesday at 17.15
- All groups
- This lab session goal is to get everyone started.
  - Visual studio code
  - Sendlab
  - Git