

# Exercise session 01

Computer Science – CSE – AS 24


25.09.2024

# Today's Schedule

Learning Objectives  
Getting to know each other  
Organizational Information  
Integer Division & Modulo  
Binary Representation  
Expressions and Evaluations  
Outro



`n.ethz.ch/~iopopa`

 [Link to Webpage](#)

 [Send an e-Mail](#)

# 1. Learning Objectives

---

# Learning Objectives

## Objectives for today

- Getting to know each other
- Organizational Information
- How to [code]expert
- Integer Division and Modulo
- Binary Representation
- Valid C++ expressions recognition
- C++ expression evaluation

## 2. Getting to know each other

---

# Your TA

## **Ioana**

- Born and raised in Romania, but always dreamed about studying abroad
- Loved sciences, but had no clue what to do in the future
- Chose CSE at ETH for its interdisciplinarity
- Had a good grade in the Informatik Exam
- Loves teaching
- Would never refuse a good coffee
- Here to help you, there are no stupid questions

# Your Turn!

## **Introduce Yourself!**

- What's your name?
- Where are you from?
- Why CSE at ETH?
- Do you have any prior programming experience?
- What is your coolest hobby?

# Questions?

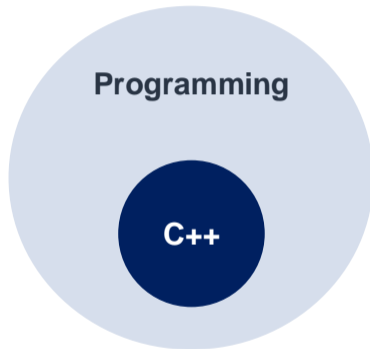


## 3. Organizational Information

---

## The goal of this course

- Lecture
- Exercise session
- Weekly exercises
- Bonus exercises
- Study Center



# Weekly Schedule

Monday

Tuesday

Wednesday

Thursday

Friday

Saturday

Sunday

Monday

Lecture  
CSE

Lecture  
IFMP

Lab  
CSE

Exercise Sessions

Study  
Center

Solving homeworks on CodeExpert  
(Released on Monday 6:00, Deadline on following Monday 18:00)

## Weekly and Bonus exercises

- All exercises are accessible on [code]expert (<https://expert.ethz.ch>).
- You first need to enroll in the class with the link you have received.
  
- **Weekly exercises:**
  - Purpose: practice the new material.
  - Released: Monday at 6:00.
  - Deadline: One week later (Monday at 18:00).
  - Allow earning experience points (XP).
  
- **Bonus exercises** (need around 2/3 experience points to unlock):
  - Purpose: combine knowledge from different topics.
  - Allow earning max +0.25 towards the final grade (with 2/3 of bonus points).

## Weekly and Bonus exercises

- All exercises are accessible on [code]expert (<https://expert.ethz.ch>).
- You first need to enroll in the class with the link you have received.

### Important!

- **Weekly exercises**

- Purpose: practice
- Released: 1 week before
- Deadline: 1 week after
- Allow earning max +0.25

While solving exercises you should **only** use the constructs that were already introduced in the lecture and are not forbidden by the task description. Note that the autograder might impose further restrictions (such as not using global variables) that might not be explicitly stated in the task description. Further note that warnings are treated as errors. Thus, check the output of the autograder carefully to avoid getting 0 points.

- **Bonus exercises**

- Purpose: combine knowledge from different topics.
- Allow earning max +0.25 towards the final grade (with 2/3 of bonus points).

Each week we will publish a summary that lists what concepts were introduced on that week.

## Exercise Sessions

- Purpose: Prepare for solving future and past exercises.
- Approach: mostly interactive classroom activities and constructive discussion.
- We expect that you will:
  - Actively participate in the classroom activities.
  - Ask questions if you do not understand what we are teaching, why we are teaching a particular topic
- Note: Making mistakes is completely normal, we are just learning. Please avoid doing things that may distract others. If a task is too easy, help others.

## Study Center

- Purpose: a chance to ask for individual help regarding the course.

### IFMP:

- Time: Wednesday 16:15-18:15, starting from September 25th
- Place: Mensa Polyterrasse
- Link: <https://studycenter.ethz.ch/>

### CSE (shared with MAVT):

- Time: Wednesday 18:15-20:00, starting from October 2nd
- Place: ETA F 5

## Info & Contact

- More information is given in the organisational information sheet.
- For questions regarding the *content* of the lecture you can ask in class.
- For questions regarding the *exercises*, you can ask your TA.
- For *administrative* questions, please contact the head TA (see website for email address).



# Questions?

# About the exercise session

- Everything is uploaded to my website (usually on the same day)
- Will not be recorded
- Please ask questions whenever you have any
- Please participate in the lesson
- Please correct me if I make any mistakes
- You can also write questions directly in your code on [code]expert, but I will only see them after you have submitted the solution:

```
int a = 5; //type your question here
```

- Questions via email are always welcome

# About this course

- It's not the hardest course...
- ...but one of the most important ones
- If you don't pass the BPB-1, it probably won't be because of Informatik 1
- Exams won't take place until January, but...
- **...PRACTICE DURING THE SEMESTER AND PERSIST**
- ...and if you get stuck: get help, e.g. from
  - me
  - the Study Center
  - your fellow classmates
  - the professors

# About [code]expert

- code expert can be a little picky, so follow the instructions very carefully (avoid unnecessary text)
- The autograder will do most of the corrections
- I will give you the last few points for style, documentation, approach, etc.
- Text tasks are corrected entirely manually
- You can expect feedback on the tasks within a week (from me) (if urgent, just send me an email)

# Questions?

## 4. Integer Division & Modulo

---

# The Difference between = and ==

## Assignment Operator (=)

Needed to assign values to variables

```
int a = 42; // assigns the value 42 to the variable a  
int b = 18; // assigns the value 18 to the variable b
```

## Equality Operator (==)

Used to check equality between variables

```
(a == b) // this "expression" will equal 1 (true)  
// or 0 (false) ("boolean")
```

# Integer Division & Modulo

## Integer Division (a/b)

The Compiler “ignores” decimal places when dividing (unsigned) int by (unsigned) int.

## Modulo (a%b)

Division with Remainder, but outputs *only* the remainder.



# Integer Division & Modulo

## Integer Division (a/b)

The Compiler “ignores” decimal places when dividing (unsigned) int by (unsigned) int.

- $7 / 3 ==$
- $15 / 4 ==$
- $16 / 4 ==$

## Modulo (a%b)

Division with Remainder, but outputs *only* the remainder.

# Integer Division & Modulo

## Integer Division (a/b)

The Compiler “ignores” decimal places when dividing (unsigned) int by (unsigned) int.

- `7 / 3 ==`
- `15 / 4 ==`
- `16 / 4 ==`

## Modulo (a%b)

Division with Remainder, but outputs *only* the remainder.

- `7 % 3 ==`
- `15 % 4 ==`
- `16 % 4 ==`

# Integer Division & Modulo

## Integer Division (a/b)

The Compiler “ignores” decimal places when dividing (unsigned) int by (unsigned) int.

- $7 / 3 ==$
- $15 / 4 ==$
- $16 / 4 ==$

## Important Identity

$$(a / b) * b + a \% b == a$$

The modulo division is unsuitable for calculating with non-integer numbers

## Modulo (a%b)

Division with Remainder, but outputs *only* the remainder.

- $7 \% 3 ==$
- $15 \% 4 ==$
- $16 \% 4 ==$

# Questions?

# Let's see what you have learned

- Go to [expert.ethz.ch](http://expert.ethz.ch)
- Log in
- Go to “Code Examples”
- Under “Week 2: Exercise Session”, open “Last Three Digits”
- Try to solve the task (10 minutes)
- We will look at your approaches later

# Exercise Prompt

## **Task - "Last Three Digits"**

Write a program which reads in an integer  $a$  larger than 1000 and outputs its last three digits with a space between them.

For example, if  $a = 14325$ , the output should be 3 2 5.

# 5. Binary Representation

---

# Binary Representation

0 1 0 0 0 1 1 0



# Binary Representation

0	1	0	0	0	1	1	0
$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
x128	x64	x32	x16	x8	x4	x2	x1

# Binary Representation

0	1	0	0	0	1	1	0
---	---	---	---	---	---	---	---

$2^7$	$2^6$	$2^5$	$2^4$	$2^3$	$2^2$	$2^1$	$2^0$
-------	-------	-------	-------	-------	-------	-------	-------

x128	x64	x32	x16	x8	x4	x2	x1
------	-----	-----	-----	----	----	----	----

64	+				4	+	2
----	---	--	--	--	---	---	---

---

70

# The clock that no one can read

What time is it?



# The clock that no one can read

What time is it?



23:06:18

# Binary Representation

## Question

What would be an algorithm that derives the binary representation of a decimal number?

For example, how would you convert 61 into binary?

# Binary Representation

## Question

What would be an algorithm that derives the binary representation of a decimal number?

For example, how would you convert 61 into binary?

## Solution

Divide the decimal number by two and keep the rest (just like a modulo division). Divide the remaining number again, and so on and so forth until you reach 0.

# Binary Representation

$$61 = 2 * 30 + 1$$

$$30 = 2 * 15 + 0$$

$$15 = 2 * 7 + 1$$

$$7 = 2 * 3 + 1$$

$$3 = 2 * 1 + 1$$

$$1 = 2 * 0 + 1$$

# Binary Representation

$$61 = 2 * 30 + 1$$

$$30 = 2 * 15 + 0$$

$$15 = 2 * 7 + 1$$

$$7 = 2 * 3 + 1$$

$$3 = 2 * 1 + 1$$

$$1 = 2 * 0 + 1$$

Then read the last column from bottom to top and you're done!

$$61_{10} = 111101_2$$



# Binary Representation of Negative Integers

## Question

But how can we store negative numbers?

*Tipp:*

$$n + (-n) = 0$$

# Binary Representation of Negative Integers

## Question

But how can we store negative numbers?

*Tipp:*

$$n + (-n) = 0$$

## Solution

Treat the leading digit as the negative of its “actual” value

# Binary Representation of Negative Integers

## Question

In 4bit binary 1 is represented as:

0001

And 0 is represented as:

0000

What do we need to add to 0001 to get 0000?

$$\begin{array}{r} 0001 \\ +???? \\ \hline 0000 \end{array}$$

# Binary Representation of Negative Integers

## Question

In 4bit binary 1 is represented as:

0001

And 0 is represented as:

0000

What do we need to add to 0001 to get 0000?

$$\begin{array}{r} 0001 \\ +???? \\ \hline 0000 \end{array}$$

**Answer:** 1111

# Binary Representation of Negative Integers

## Question

How do you compute the (signed) `int` representation of a negative integer  $n < 0$ ?

# Binary Representation of Negative Integers

## Question

How do you compute the (signed) `int` representation of a negative integer  $n < 0$ ?

## Solution

1. Convert the absolute value of  $x$  to binary.
2. Flip bits.
3. Add 1.

# Questions?

## 6. Expressions and Evaluations

---



# What are expressions?

## Expressions

- This is an expression:  $5u + 5 * 3u$
- Expressions are evaluated piece by piece
- Pay attention to the precedence of the operators (multiplication before addition) and the data types (more on that later)

# Evaluate following expression:

$$5u + 5 * 3u$$

What does  $u$  stand for? What do we do first?

# Evaluate following expression:

`5u + 5 * 3u`

What does `u` stand for? What do we do first?

`5u + (5 * 3u)` (precedence of operators)

`5u + 15u` (the result of the multiplication maintains the data type `unsigned int`)

# Evaluate following expression:

`5u + 5 * 3u`

What does `u` stand for? What do we do first?

`5u + (5 * 3u)` (precedence of operators)

`5u + 15u` (the result of the multiplication maintains the data type `unsigned int`)

`20u` (basic addition)

# Recognizing valid expressions

**Which of the following snippets is a valid C++ expression?**

1. `1 * (2 * 3)`

2. `(a = 1)`

3. `(1`

4. `(a * 3) = (b * 5)`

# Recognizing valid expressions

## Which of the following snippets is a valid C++ expression?

1. `1 * (2 * 3)`
2. `(a = 1)`
3. `(1`
4. `(a * 3) = (b * 5)`

### Solution:

- 3. is not valid because the bracket ( is not closed
- 4. is invalid, since `(a * 3)` is an r-value, but the left operand of the assignment operator must be an l-value
- 1. and 2. are valid C++ expressions

# Recognizing valid expressions

**Which of the following snippets is an l-value and which is an r-value?**

1.  $1 * (2 * 3)$

2.  $(a = 1)$

3.  $(1$

4.  $(a * 3) = (b * 5)$

# Recognizing valid expressions

**Which of the following snippets is an l-value and which is an r-value?**

1.  $1 * (2 * 3)$
2.  $(a = 1)$
3.  $(1)$
4.  $(a * 3) = (b * 5)$

**Solution:**

- 3. and 4. are invalid expressions
- 1. is an **r-value** by definition of the multiplication operator  $*$
- 2. is an **l-value** by definition of the assignment operator  $=$



# Recognizing valid expressions

**Determine the values of the expressions and explain how these values are obtained.**

1.  $1 * (2 * 3)$

2.  $(a = 1)$

3.  $(1$

4.  $(a * 3) = (b * 5)$

# Recognizing valid expressions

**Determine the values of the expressions and explain how these values are obtained.**

1.  $1 * (2 * 3)$

2.  $(a = 1)$

3.  $(1$

4.  $(a * 3) = (b * 5)$

**Solution:**

- 3. and 4. are invalid expressions, so they cannot be computed
- The value of 1. is 6, obtained by evaluating the two multiplications
- The value of 2. is 1, obtained by assigning value 1 to a (a is returned)

# Questions?

## 7. Outro

---

# General Questions?

# General Questions?

See you next time!