Applied Programming I

Programming computers in a nutshell.

EXAM

Something to consider...

General remarks

The exam is a written exam.

- You need to bring your computer.
- You write the answers to the problems with your computer.
- You submit your answers digitally.

- Course-specific note: You are allowed to use the itsLearning resources of this course.

General remarks: Plan

- Look at all the problems and the distribution of achievable points.
- Consider planning the time you spend on each problem in relation to the maximum score for that problem:
    - This means that it might be a good idea to spend more time on the problems that are worth more points.
- Consider solving the easy problems first so that you do not miss them when you run out of time:
    - This means: Pick the low-hanging fruit first.
- Remember that the examiner cannot award any points for a blank answer/no solution.

General remarks: Read

1. Read the problem / task carefully.
2. Before starting with the problem, read the problem / task carefully.
3. After answering the problem, read the problem / task carefully again and ensure you covered all.

General remarks: Don’t panic

- On the first reading, problems might seem overwhelming.
- Calm down and read the problem again.
- Then apply ‘**divide and conquer**’:
    - Identify the subproblems and solve subproblem after subproblem.
    - If one subproblem does not seem accessible on the first thought, continue with the next one. Go back later.
    - Each problem has stated subproblems.
- Follow the instructions. Use the hints.

Specific remarks

Course work load

- ECTS: 5
- This corresponds to approximately 140 hours student work
  - Of these only 48 hours in class,
  - And 92 hours self-study (work with the learning materials, repeating, assignments, exam preparation)

Examination

- Written examination
- 2-hour duration
- Grading: Pass/Fail (you pass with achieving half of the points)
- Extra credits (achievable by the assignments) will be maximum of 10% of final grade and these will contribute to the pass rate.
- The form of examination in the re-examination is the same as in the ordinary examination.

Preparation

- You will have access to the slides in **itsLearning** during the exam.
- Become familiar with the material that you can use during the exam.
- Look at the course resources in itsLearning. See what you have at your disposal during the exam so that you can focus on learning.
- Use the questions on the next slides as orientation for preparation.
- You are allowed to bring your own notes into the exam.

No internet / AI in exam

- There is no AI or internet access during the exam.
- If you understand the principles on the slides regarding the exam, you do not need AI during the exam.
- Remark about AI in the course:
    - AI was encouraged to be used (to make your life easier and not to stop you from thinking) but the tasks were designed to be solvable without AI.
    - You should use it as support tool, because it offers a lot of help.
    - However, you need to understand and verify the answers from AI tools.

Core messages of the course

1. The course is **not** about learning programming **languages but** about understanding that programming **paradigms** are the recurring higher-level concepts of programming languages. Two important ones are:
    - Imperative programming
    - Object-oriented programming

   Use these paradigms to gain an understanding of a programming language.
2. The remainder of the course should provide you with starting points to find the right tools for application programming.

Content excluded in exam

- This topics of lecture 11 and 12 are **excluded** in the exam:
  - Data analysis in Python
  - Game programming in Unity/C#

Could you solve this in RISC-V Assembly?

If you have all the assembly instruction at hand, could you implement

- Variables (load and store from memory – what is an address?)
- Condition (compare and branch)
- Iteration (use a counter within a condition)

And explain how you have done it?

Solve this problem – without Copilot.

- You have the following RISC-V assembly instructions:
    - Branch less or equal: **ble** rs1, rs2, label: branch to label if rs1 <= rs2.
    - Load immediate: **li** rd, #imm: rd gets the value imm
    - Add immediate: **addi** rd, #imm: rd will be incremented by imm
- Write an iteration from 0 to 9 in RISC-V assembly.

Solve this problem - without Copilot.

- You have the following RISC-V on the right hand side
- Explain what the code does and what the value of register *t2* will be.
- Note: You find an instruction summary in itslearning (lecture 1)

```
.data           # data section
nbs:   .space 8   # space for 8 bytes
.text           # program section
  li t0, 2        # li (load immediate)
  la t1, nbs      # la (load address)
  sb t0, 0(t1).   # sb (store byte)
  li t0, 3
  sb t0, 1(t1)
  addi t1, t1, 1  # addi (add immediate)
  lb t2, 0(t1)    # lb (load byte)
  add t2, t2, t0  # add
```

Could you do this with a programming language used in the course?

Using imperative programming, could implement

- Use of variables of different types,
- Functions with parameters and return value,
- Conditions comparing variables,
- Iterations with an iteration variable,

and explain what you have done?

Could you explain source code that is given to you?

Solve this problem – without Copilot.

- Write a function *isVowel* that takes as parameter a character and returns a Boolean: true if the character is a vowel (a, e, i, o, u), otherwise false.
- Use this function and write program that counts the number of vowels in a string (array of characters).

Solve this problem – without Copilot.

- You have the following Python code.
- Explain what happening and how the variables change after each iteration.
- What are the final values of x and y?

```
x = 1
y = 4
z = 3
for i in range(6): # iterating i: 0,..,5
 if x == y:
  y = z
 else:
  x = 2*x
  y = y+1
x = x + 1
```

Could you do this with a programming language used in the course?

Using object-oriented programming, could you implement

- Class with methods and attributes,
- Derived classes,
- Instantiate class objects and use them,

and explain how you do it if you are provided with a description of a problem?

Solve this problem – without Copilot.

- Implement the class *Load*. The class *Load* has two private attributes:
  - *maxLoad* of type int,
  - *curLoad* of type int.
- and the public methods:
  - *constructor(max: int)*: takes the parameter *max* and sets the *maxLoad* to *max*.
  - *addLoad(load: int): bool:* takes the parameter *load*. It checks if *load+curLoad* is lower than *maxLoad*. If so, it adds load to *curLoad and r*eturns true. Otherwise it returns false.

```
+---------------------------+
|     Load                |
+---------------------------+
| - maxLoad: int           |
| - curLoad: int           |
+---------------------------+
| + Load(max: int)         |
| + addLoad(load: int): bool |
+---------------------------+
```

Solve this problem – without Copilot.

Implement the three classes:

- Class *Tree* has the public attribute *height* of type *float* and no methods.
- Class *DeciduousTree* is derived from the class *Tree* and has the public attribute *leafType* of type *int* and no methods.
- Class Conifer is derived from the class Tree and has the public attribute *needleLength* of type *float* and no methods.

```
            +-----------------+
            |     Tree      |
            +-----------------+
            | + height: float |
            +-----------------+
               ^          ^
               |          |
+----------------------+  +----------------------+
|   Conifer            |  |   DeciduousTree      |
+----------------------+  +----------------------+
| + needleLength: float |  | + leafType: int      |
+----------------------+  +----------------------+
```

Could you do this within embedded system programming?

Assuming all documentation is available, could you

- Describe the steps how to use the API required to solve a given problem?
- Understand a program (on a microcontroller) that uses a HAL or RTOS and explain what it does?
- Explain how a program (on a microcontroller) should communicate with a connected device (e.g., sensor)?

Solve this problem – without Copilot.

- You have a sensor communicating with I2C and are going to connect it with your microcontroller. The sensor measures the brightness level.
- What information do you need to lookup in the data sheet? How do you use the information?

Solve this problem – without Copilot.

You have the FreeRTOS code. There are two tasks. Explain how the tasks are scheduled – when does a task get/use the CPU over time. What role do the tasks’ priorities have and how are these?

```
void vTask1( void *pvParameters ) {
  TickType_t xLastWakeTime;
  xLastWakeTime = xTaskGetTickCount();
  for( ;; ) {
    ...
    vTaskDelayUntil( &xLastWakeTime, pdMS_TO_TICKS( 1000 ) );
  } }
void vTask2( void *pvParameters ) {
  for( ;; ) {
    ...
  } }
 void app_main( void ) {
  xTaskCreate( vTask1, "Task 1", 4096, NULL, 3, NULL );
  xTaskCreate( vTask2, "Task 2", 4096, NULL, 1, NULL ); }
```

Could you do this with a website?

- Explain the basic components and structure of a HTML website?
- Write a JavaScript program within a HTML site to get data from the website using DOM and update the website? And explain it.

Solve this problem – without Copilot.

- You have the following HTML code of a webpage that shall check if an integer number is a multiple of seven. The result shall be printed on the bottom of the page.
- Implement (write code with comments) the missing JavaScript function isMultipleOf7().

```html
<!DOCTYPE html>
<html>
<head>
<title>Is a multiple of 7 checker</title>
<script src=”ismo7.js" defer></script>
</head>
<body>
<h1>Check number if multiple of 7</h1>
<label for="decimalInput">Enter a Decimal Number:</label>
<input type="number" id="decimalInput">
<button onclick=”isMultipleOf7()">Check</button>
<p id="result"></p>
</body>
</html>
```

Check your solutions

1. Discuss your solutions with your peers.

2. Type in the problem and your solution into Copilot. Let Copilot help you to check and explain what you have done right or not.

Evaluation

Please take part. We will discuss intermediate results next week.

Valuable comments 1/2

Maybe you could comment on some of these points:

- Tasks in the lecture
  - Did you work on the **tasks** during the lecture (and after the lecture)? Were they helpful? How was the teachers’ support solving the tasks?
- Assignments
  - Were the **assignments** reasonable? Did they support learning?
- Classroom teaching
  - Did you follow along during **classroom** teaching? Did you actively participate or prefer to listen only? Were questions answered properly in discussions?

Valuable comments 2/2

- Material:
  - Were the **slides** useful? Did you use them for preparation or review?
  - Was the **code** on GitHub useful? Did you work with it outside the lectures?
- Overall :
  - How much **time** did you spend each week for the course? (be honest)
  - What **topics** did you like, understand, did not like, did not understand?
- And anything else:
  - What was good or bad that should be mentioned?

Break

15 minutes : Please fill out the evaluation if have not yet done.