
WELCOME
TechPro

[Nasos Lentzas]

[Basic Programming Concepts]

[09/10/2024]

TechPro

In this Course

1. Problem solving
2. Introduction to programming logic
3. Algorithms
4. Pseudocode
5. Logical operators

TechPro

Discussion

{Programming Experience}

TechPro

Problem solving – Define the problem

Diagnose the situation so that your focus is on the problem, not just its symptoms. Review and document how processes currently work. Evaluate the possible impact of new tools and revise policies in the development of your model.

- Differentiate fact from opinion
- Specify underlying causes
- Consult each faction involved for information
- State the problem specifically
- Identify what standard or expectation is violated
- Determine in which process the problem lies
- Avoid trying to solve the problem without data

TechPro

Problem solving – Generate solutions

Postpone the selection of one solution until several problem-solving alternatives have been proposed. Considering multiple alternatives can significantly enhance the value of your ideal solution. Once you have decided on the "what should be" model, this target standard becomes the basis for developing a road map for investigating alternatives.

- Postpone evaluating alternatives initially
- Include all involved individuals in the generating of alternatives
- Specify alternatives consistent with organizational goals
- Specify short- and long-term alternatives
- Brainstorm on others' ideas
- Seek alternatives that may solve the problem

TechPro

Problem solving – Evaluate & select solution

- A particular alternative will solve the problem without causing other unanticipated problems.
- All the individuals involved will accept the alternative.
- Implementation of the alternative is likely.
- The alternative fits within the organizational constraints.
- Evaluate all alternatives without bias
- Evaluate both proven and possible outcomes
- State the selected alternative explicitly

TechPro

Problem solving – Implementation

Solution implementations is usually done by involving all team members. Involving others in the implementation is an effective way to gain buy-in and support and minimize resistance to subsequent changes. Feedback channels should be built into the implementation. This allows for continuous monitoring and testing of actual events against expectations.

- Plan and implement a pilot test of the chosen alternative
- Gather feedback from all affected parties
- Seek acceptance or consensus by all those affected
- Establish ongoing measures and monitoring
- Evaluate long-term results based on final solution

TechPro

Discussion

How would you troubleshoot an error in a software product that has been released to customers?

TechPro

Survey

{Programming vs Coding}

TechPro

Programming Logic

The basic way programmers understand and organize their code to produce desired results is known as programming logic. It involves decomposing issues into smaller, more manageable components and formulating a plan of action to address each one.

- Sequence: arranging commands in a sequential order, allowing the computer to execute them one after another
- Selection: Conditions or logical tests are used to direct the flow of a program.
- Iteration (Loops): Loops enable the repetition of certain tasks until a condition is met. This helps in automating repetitive tasks and managing data efficiently.

TechPro

Programming Logic

Why is Programming Logic Important?

1. Problem Solving: Break down complex problems into smaller, manageable parts. This simplification enables developers to solve problems systematically.
2. Efficient Code Writing: using logical structures, programmers can write code that is not only understandable but also efficient.
3. Enhancing Debugging Skills: Logical thinking assists in identifying errors within the code and debugging more effectively.

TechPro

Discussion

{Can you tell me how to make pasta?}

Algorithms

An algorithm is a set of commands that must be followed for a computer to perform calculations or other problem-solving operations. An algorithm is a finite set of instructions carried out in a specific order to perform a particular task. It is not the entire program or code; it is simple logic to a problem represented as an informal description in the form of a flowchart or pseudocode.

- Problem: A problem can be defined as a real-world problem or real-world instance problem for which you need to develop a program or set of instructions
- Algorithm: An algorithm is defined as a step-by-step process that will be designed for a problem.
- Input: After designing an algorithm, the algorithm is given the necessary and desired inputs.
- Output: The outcome or result of the program is referred to as the output.

TechPro

Algorithms

An algorithm is a set of commands that must be followed for a computer to perform calculations or other problem-solving operations. An algorithm is a finite set of instructions carried out in a specific order to perform a particular task. It is not the entire program or code; it is simple logic to a problem represented as an informal description in the form of a flowchart or pseudocode.

- Problem: A problem can be defined as a real-world problem or real-world instance problem for which you need to develop a program or set of instructions
- Algorithm: An algorithm is defined as a step-by-step process that will be designed for a problem.
- Input: After designing an algorithm, the algorithm is given the necessary and desired inputs.
- Output: The outcome or result of the program is referred to as the output.
- Efficiency: A key aspect of algorithms is their efficiency, aiming to accomplish tasks quickly and with minimal resources.
- Optimization: Algorithm designers constantly seek ways to optimize their algorithms, making them faster and more reliable.

TechPro

Characteristics of an Algorithm

- **Finiteness**: An algorithm must always have a finite number of steps before it ends. When the operation is finished, it must have a defined endpoint or output.
- **Definiteness**: An algorithm needs to have exact definitions for each step. Clear and straightforward directions ensure that every step is understood and can be taken easily.
- **Input**: An algorithm requires one or more inputs.
- **Output**: One or more outputs must be produced by an algorithm. The output is the outcome of the algorithm after every step has been completed.
- **Effectiveness**: An algorithm's stages must be sufficiently straightforward to be carried out in a finite time utilizing fundamental operations
- **Generality**: Rather than being limited to a single particular case, an algorithm should be able to solve a group of issues.

TechPro

Factors of an Algorithm

- **Modularity**: This feature was perfectly designed for the algorithm if you are given a problem and break it down into small-small modules or small-small steps
- **Correctness**: An algorithm's correctness is defined as when the given inputs produce the desired output, indicating that the algorithm was designed correctly
- **Maintainability**: It means that the algorithm should be designed in a straightforward, structured way so that when you redefine the algorithm, no significant changes are made to the algorithm
- **Functionality**: It considers various logical steps to solve a real-world problem.
- **Robustness**: Robustness refers to an algorithm's ability to define your problem clearly.
- **User-friendly**: If the algorithm is difficult to understand, the designer will not explain it to the programmer.
- **Simplicity**: If an algorithm is simple, it is simple to understand.
- **Extensibility**: Your algorithm should be extensible if another algorithm designer or programmer wants to use it.

Qualities of a Good Algorithm

- **Efficiency**: A good algorithm should perform its task quickly and use minimal resources.
- **Correctness**: It must produce the correct and accurate output for all valid inputs.
- **Clarity**: The algorithm should be easy to understand and comprehend, making it maintainable and modifiable.
- **Scalability**: It should handle larger data sets and problem sizes without a significant decrease in performance.
- **Reliability**: The algorithm should consistently deliver correct results under different conditions and environments.
- **Optimality**: Striving for the most efficient solution within the given problem constraints.
- **Robustness**: Capable of handling unexpected inputs or errors gracefully without crashing.
- **Adaptability**: Ideally, it can be applied to a range of related problems with minimal adjustments.
- **Simplicity**: Keeping the algorithm as simple as possible while meeting its requirements, avoiding unnecessary complexity.

Types of algorithms

- **Brute Force Algorithm**: A straightforward approach that exhaustively tries all possible solutions, suitable for small problem instances.
- **Recursive Algorithm**: A method that breaks a problem into smaller, similar subproblems and repeatedly applies itself to solve them until reaching a base case
- **Encryption Algorithm**: Utilized to transform data into a secure, unreadable form using cryptographic techniques
- **Backtracking Algorithm**: A trial-and-error technique used to explore potential solutions by undoing choices when they lead to an incorrect outcome
- **Searching Algorithm**: Designed to find a specific target within a dataset, enabling efficient retrieval of information from sorted or unsorted collections.
- **Sorting Algorithm**: Aimed at arranging elements in a specific order, like numerical or alphabetical, to enhance data organization and retrieval.
- **Hashing Algorithm**: Converts data into a fixed-size hash value, enabling rapid data access and retrieval in hash tables, commonly used in databases and password storage.
- **Divide and Conquer Algorithm**: Breaks a complex problem into smaller subproblems, solves them independently, and then combines their solutions to address the original problem effectively.
- **Greedy Algorithm**: Makes locally optimal choices at each step in the hope of finding a global optimum, useful for optimization problems but may not always lead to the best solution.
- **Randomized Algorithm**: Utilizes randomness in its steps to achieve a solution, often used in situations where an approximate or probabilistic answer suffices.
- **Dynamic Programming Algorithm**: Stores and reuses intermediate results to avoid redundant computations, enhancing the efficiency of solving complex problems.

TechPro

Pseudocode

Pseudocode is a technique used to describe the distinct steps of an algorithm in a manner that’s easy to understand for anyone with basic programming knowledge. Although pseudocode is a syntax-free description of an algorithm, it must provide a full description of the algorithm’s logic so that moving from pseudocode to implementation is merely a task of translating each line into code using the syntax of any given programming language.

How to Write Pseudocode

Everyone has their own style of presenting since humans are reading it and not a computer; pseudocode's rules are less rigorous than those of a programming language

- Always capitalize the initial word
- Make only one statement per line.
- Indent to show hierarchy, improve readability and show nested constructs.
- Always end multi-line sections using any of the END keywords
- Keep your statements programming language independent.
- Use the naming domain of the problem, not that of the implementation. For instance: "Append the last name to the first name" instead of "name = first+last."
- Keep it simple, concise and readable.

Why Use Pseudocode?

- Pseudocode Is Easier to Read: Makes communicating between different specialties easier and more efficient.
- Pseudocode Simplifies Code Construction: Converting pseudocode into real code written in any programming language is much easier and faster.
- Pseudocode Is a Helpful Starting Point for Documentation
- Pseudocode Allows for Quick Bug Detection: Pseudocode is written in a human-readable format, it is easier to edit and discover bugs before writing a single line of code

TechPro

Operators

In computer programming, operators are constructs defined within programming languages which behave generally like functions, but which differ syntactically or semantically.

- Arithmetic operators that perform arithmetic operations with numeric operands:
  - Unary ++ (increment), -- (decrement), + (plus), and - (minus) operators
  - Binary * (multiplication), / (division), % (remainder), + (addition), and - (subtraction) operators
- Comparison operators that compare numeric operands: < (less than), > (more than), <= (less than or equal), >= (more than or equal)
- Equality operators: == (equal), != (not equal)
- Logical operators: AND (&&), OR (||) , NOT (!), XOR (^)
- Bitwise operators:
  - Bitwise complement operator ~
  - Left-shift operator <<
  - Right-shift operator >>
  - Logical AND operator &
  - Logical OR operator |

TechPro

Logical Operators

Input		Output		
A	B	AND	OR	XOR
0	0	0	0	0
0	1	0	1	1
1	0	0	1	1
1	1	1	1	0

Bitwise operators

Operates at the level of the individual bits. On simple low-cost processors, typically, bitwise operations are substantially faster than division, several times faster than multiplication, and sometimes significantly faster than addition.

- NOT (~): logical negation e.g: ~ 0111 = 1000
- AND (&): performs the logical AND operation on each pair of the corresponding bits: 0101 & 0011 = 0001
- OR (|): performs the logical OR operation on each pair of the corresponding bits: 0101 | 0011 = 0110
- XOR(^): performs the XOR operation on each pair of the corresponding bits
- LEFT SHIFT (<<) / RIGHT SHIFT (>>): logical shift of the bits left or right by x places: 0101 << 1 = 1010, 1001 >> 1 = 0100

TechPro

DIV / MOD

- Integer division operation (DIV) calculates the quotient
- Modulus operation calculates the remainder
- Both expressions work on integer values

E.G:

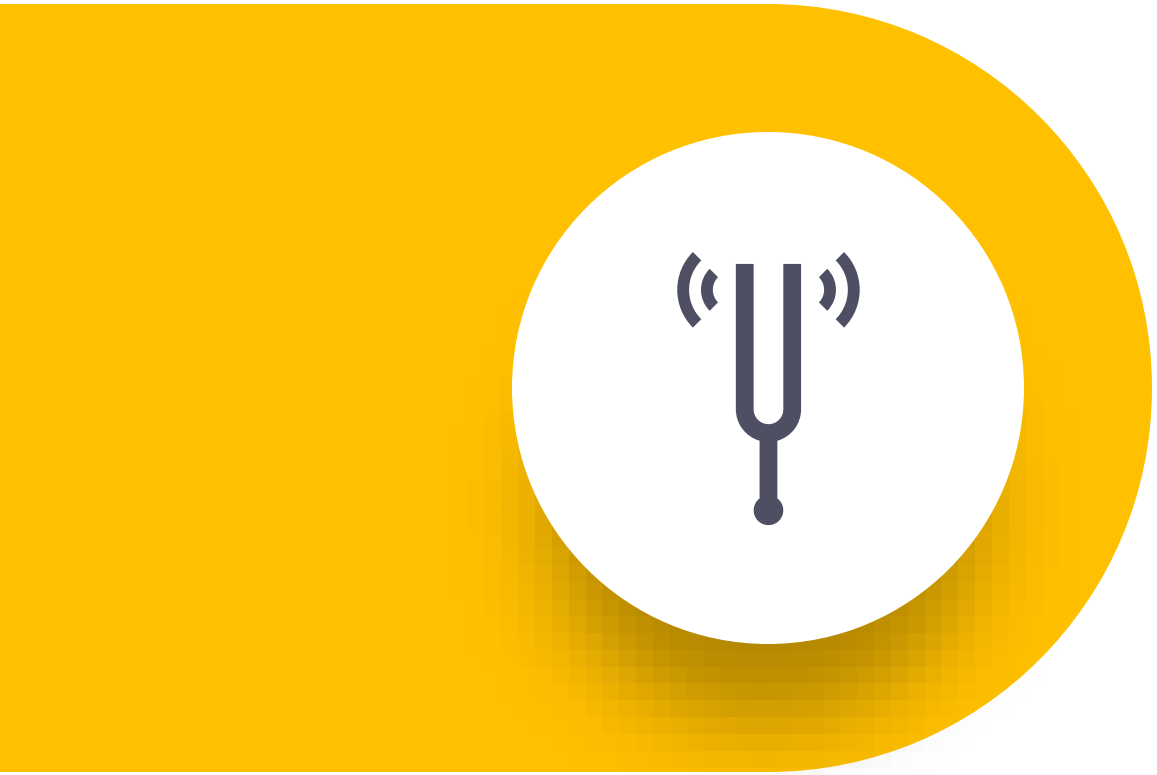
- 17 DIV 5 = 3, 17 MOD 5 = 2
- -17 DIV 5 = -3, -17 MOD 5 = -2
- 128 DIV 10 = 12, 128 MOD 10 = 8

TechPro

Quiz

Given two variables, x, and y, swap the two variables without using a third variable.

TechPro



Homework

Download:
https://alkisg.mysch.gr/downloads/

Familiarize yourself with the IDE.

Feedback Discussion

TechPro

Thank you

TechPro