Session #3

Data Types, Scopes, and Control Flow

ICPC SCU

Session Structure

- Data Types in C++
- Scopes
- namespaces
- Indentation
- Conditionals and Control Flow
- Conditional Structures
- Let’s start coding

Data Types in C++

- Data Types specify the kind of data a variable can hold, like numbers, text, or true/false, values.
- Data Types help C++ understand how to store and process different kinds of information, ensuring that the right operations can be performed on them.
- Most important -for CPers- data types:
    - int
        - ‘int’ is short for “integer”
        - It’s used to store whole numbers (positive or negative)
    - long long int
        - ‘long long int’ is used for very large whole numbers
    - float
        - float is for decimal numbers. It’s suitable for values with fractional parts
    - double
        - double is like float but can store more significant digits
        - It’s often used when high precision is needed
    - char
        - ‘char’ stands for “character”
        - It’s used to store single characters, like letters or symbols
    - bool
        - bool represents Boolean values
        - It can be either true or false and is used for logical decisions
    - Task: Search for more data types, compare between different data types, and their limitation

Scopes

- Scopes determine where in your code a variable is accessible
- Variables can be local (only usable in a specific part of the code or global (accessible throughout the program), depending on their scope
- Let’s see some examples.

Namespaces

- A namespace in C++ is a feature that allows you to organize your code into distinct and separate scopes, preventing naming conflicts and providing a way to group related code elements
- Search for namespaces in C++

Cin and cout Streams

- cin and cout are C++ streams used for input and output operations, respectively. They provide a way to interact with the console and handle data entering or leaving your program.
- Search for Streams in C++

Indentation

- Indentation is the use of spaces or tabs to visually organize code
- It makes code more readable by showing the structure hierarchy of the code, making it easier to understand

Conditionals and Control Flow

- Conditionals are used to make decisions in your program, and control flow guides the order of execution
- Conditionals (if, else) let you choose between different actions based on conditions, and control flow helps dictate the sequence in which these actions occur.

Here is the rather trivial schematic representation of straight-line control flow:

CONDITIONAL EXECUTION

Not all programs are straight roads. We may, for example, want to create a branching road, where the program takes the proper branch based on the situation at hand. This is called *conditional execution*.

Conditional Structures (if, if else, else)

- These are ways to create conditional statements in C++
- ‘If’ checks a condition, ‘else if’ allows for multiple conditions, and ‘else’ specifies what to do if none of the conditions are met

Tasks

- Try to add the usage of what you've learned (variables, data types, control flow, input/output streams) to your solution of the first task, how would it be?
- Describe the steps of an algorithm that takes student's score and define if the student passes the exam or not. Think carefully of the steps, and describe your thoughts clearly
- Write a simple algorithm that uses the tools that you've learned to take a temperature in Fahrenheit and Convert it to Celsius, then ask him if he wants to revert the Fahrenheit, the user should respond with y/n, depending on his answer, do your job.