PÉCSI TUDOMÁNYEGYETEM
TERMÉSZETTUDOMÁNYI KAR

“Methodology of Programming I”

Spring 2025

PTE

Zahra Ahmadipour

Zahra@gamma.ttk.pte.hu

Type Conversion

Type conversion is when you assign a value of one primitive data type to another type.

- **Widening Conversion**: Converts a value to a type that can include at least approximations of all of the values of the original type.

- **Narrowing Conversion**: Converts a value to a type that cannot store even approximations of all of the values of the original type.

Most languages provide some capability for doing explicit conversions, both widening and narrowing. In some cases, warning messages are produced when an explicit narrowing conversion results in a significant change to the value of the object being converted.

In the C-based languages, explicit type conversions are called **casts**. To specify a cast, the desired type is placed in parentheses just before the expression to be converted, as in *(int) name*.

One of the reasons for the parentheses around the type name in these conversions is that the first of these languages, C, has several two-word type names, such as:

long int.

In Java:

Widening Casting:

Done automatically.

Converting smaller type to a larger type

byte > short > int > long > float > double

Narrowing Casting:

Done by placing the type, in parentheses in front of the value.

Converting larger type to a smaller type

double > float > long > int > short > byte

Wrapper Classes

Wrapper classes provide a way to use primitive data types as objects.

primitive types and their equivalent wrapper classes:

Data type		Wrapper class	Data type		Wrapper class
byte	>>	Byte	short	>>	Short
int	>>	Integer	long	>>	Long
float	>>	Float	double	>>	Double
boolean	>>	Boolean	char	>>	Character

Operators

Based on operators' functionalities they can be classified as follows:

- Arithmetic operators
- Assignment operators
- Comparison operators
- Logical operators
- Bitwise operators

Arithmetic operators:

To perform common mathematical operations.

+	Addition	
-	Subtraction	
*	Multiplication	
/	Division:	
%	Modulus:	Division remainder
++	Increment:	Increases the value by one, prefix or postfix
--	Decrement:	Decreases the value by one, prefix or postfix

Arithmetic operators are often used for more than one purpose.

For example, **+** usually is used to specify integer addition and floating-point addition. Some languages like Java, also use it for string catenation. This multiple use of an operator is called **operator overloading** and is generally thought to be acceptable, as long as neither readability nor reliability suffers.

In programming languages, arithmetic expressions consist of operators, operands, parentheses, and function calls.

An operator can be **unary**, meaning it has a single operand, **binary**, meaning it has two operands, or **ternary**, meaning it has three operands.

Assignment operators:

To assign values to variables.

=	Assignment operator
+=	Addition assignment
-=	Subtraction assignment
*=	Multiplication assignment
/=	Division assignment

Comparison/Relational operators:

A relational operator is an operator that compares the values of its two operands. The value of a relational expression is Boolean, except when Boolean is not a type included in the language.

==	Equal to
!=	Not equal
>	Greater than
<	Less than
>=	Greater or equal
<=	less or equal

The syntax of the relational operators for equality and inequality differs among some programming languages. For example, for inequality, the C-based languages use != JavaScript and PHP have two additional relational operators,

=== and !==.

The relational operators always have lower precedence than the arithmetic operators, so that in expressions such as

*a + 1 > 2 * b*

the arithmetic expressions are evaluated first.

Logical/Boolean operators:

To determine the logic between variables and test for true or false values.

&&	Logical and	true if both statements are true
||	Logical or	true if at least one of the statements is true
!	Logical not	reverses the result

Math Class

To perform additional mathematical tasks on number, Math class's methods can be used.

Math.max(x, y)	Returns the greater of two values
Math.min(x, y)	Returns the lower of two values
Math.sqrt(x)	Returns the square root of x
Math.abs(x)	Returns the absolute(positive) value of x
Math.random()	Return a 0.0<= random number <1.0

Control Statements

Computations in imperative-language programs are accomplished by evaluating expressions and assigning the resulting values to variables.

At least two additional linguistic mechanisms are necessary to make the computations in programs flexible and powerful: some means of selecting among alternative control flow paths (of statement execution) and some means of causing the repeated execution of statements or sequences of statements. Statements that provide these kinds of capabilities are called control statements.

A **control structure** is a control statement and the collection of statements whose execution it controls.

Conditions

if…else

Use the if statement to specify a block of Java code to be executed, only if a condition is true.

Use else if statement to add a new condition, so the block of code will be executed if the previous conditions were false. We can have multiple if else statements.

Use else statement to specify a block to be executed if all previous conditions were false.

Ternary Operator

It can be used to replace multiple lines of code with a single line, and is most often used to replace simple if else statements.

It is known as the ternary operator because it consists of three operands.

Syntax:

```
variable = (condition) ? valueToBeIfTrue :  valueToBeIfFalse;
```

switch

There will be one switch expression and several case values.

The block of code where the case’s value matches the value of the switch expression, will get executed.

break keyword is used to jump out of the switch block, after a match found.

default keyword is used when none of the cases were a match for the switch expression.

Loops

An iterative statement is one that causes a statement or collection of statements to be executed zero, one, or more times. An iterative statement is often called a loop.

The first iterative statements in programming languages were directly related to arrays. This resulted from the fact that in the earliest years of computers, computing was largely numerical in nature, frequently using loops to process data in arrays.

Several categories of iteration control statements have been developed.

The primary categories are defined by how designers answered two basic design questions:

- How is the iteration controlled?
- Where should the control mechanism appear in the loop statement?

Loops can execute a block of code as long as a specified condition is reached.

Loops are handy because they save time, reduce errors, and they make code more readable.

while

The while loop, checks if the condition is true, and loops through a block of code as long as the condition remains true.

In the while loop, if there is a variable used in the condition, it always has to be increased or decreased, so the code won’t run forever.

do...while

The do…while loop is a variant of while loop.

It executes the code block once and then starts checking if the condition is true to execute it again as long as the condition is true.

In the do…while loop, if there is a variable used in the condition, it always has to be increased or decreased, so the code won’t run forever.

for

When you know exactly how many times you want to loop through a block of code, use the for loop instead of a while loop.

In a for loop there will be three statements:

1. Set a variable before the loop starts
2. Define a condition for the loop to run if it’s true
3. Increasing or decreasing the value after each execution

for each

The "for-each" loop, simplifies iterating through the values in an array or objects in a collection that implements the Iterable interface (All of the predefined generic collections in Java implement Iterable).

---

break statement can be used in all kinds of loops to jump out of them.

continue statement can be used in all kinds of loops to skip a certain part of the iteration.

References:

Concepts of Programming Languages By Robert W. Sebesta

https://docs.oracle.com/javase/tutorial/tutorialLearningPaths.html

https://www.w3schools.com/

https://www.geeksforgeeks.org/