Lecture 4

- Covers
  - Program components
  - How to design algorithms to solve problems
  - How to convert algorithms into programs
  - How to edit, compile and run Java programs
- Reading: Savitch 1.3

- Elements of a program

Program components

- Programs in all paradigms share common features
  - Input and output
  - Variables
  - Identifiers
  - Reserved words (keywords)
  - Statements
  - Comments

Input and output

- Input
  - A way of receiving information from the outside
  - Keyboard, files, devices
  - When starting the program or during the program
- Output
  - A way of sending information to the outside
  - Monitor, files, devices

Variables

- Store some data
- Are declared inside methods (or functions)
- The value stored may change as the program progresses
- In Java:
  - The data stored can be a value or a reference to an object

Identifiers

- Identifiers are names for classes, attributes, methods, variables
- In Java:
  - An identifier is made of letters, digits, $ and _
  - It must not begin with a digit
  - It must not be a reserved word (keyword)

Reserved words (keywords)

- Keywords have specific pre-defined meanings
- They cannot be used for other purposes
- In Java:
  - “import” “public” “class” are Java reserved words (keywords)
  - Java has 48 reserved words (listed in the next slide)

Reserved words (keywords)

abstract	boolean	break	byte	case
catch	char	class	const	continue
default	do	double	else	extends
final	finally	float	for	goto
if	implements	import	instanceof	int
interface	long	native	new	package
private	protected	public	return	short
static	strictftp	super	switch	synchronized
this	throw	throws	transient	try
void	volatile	while		

Statements

- A statement specifies an action
- In Java:
  - It must terminate with a semicolon
  - Examples

```
int sum;
int sum = 0;
int sum = n1 + n2;
int n1 = keyboard.nextInt( );
System.out.println("hello");
```

Comments

- Comments are ignored by the computer but written in a program to explain to the reader what the program is doing
- In Java:
  - They come in two forms: block comments and line comments
  - A block comment is enclosed between a /* and a */ and may extend over more than one line
  - A line comment starts from double slashes // and continues to the end of the line

White space

- Blanks, tabs, and new line characters are called white space characters
- Except when white space is used to separate keywords and identifiers, it is ignored by the compiler
- White space can be used to make programs easy to read
- Two main uses of white space
  - (1) indentation
  - (2) blank lines to separate parts of programs

- Execution Starting Point

Execution starting point

- Every program needs a starting point at which to start executing

Execution staring point in Java

- Java programs are a collection of classes
- Each class must be stored in a file with the same name, but also with the .java extension
- A class may have the special class method main( ), which contains instructions to start a program
- The starting point of a program must be a main method specified to the interpreter:

  > java MyClass

  starts at the main method in the class MyClass

Execution staring point in Java

- Sometimes we create classes simply to give us a place to start in the program
- We call these classes launcher classes or driver classes

- How to solve problems

(Using Algorithms)

Programs and algorithms

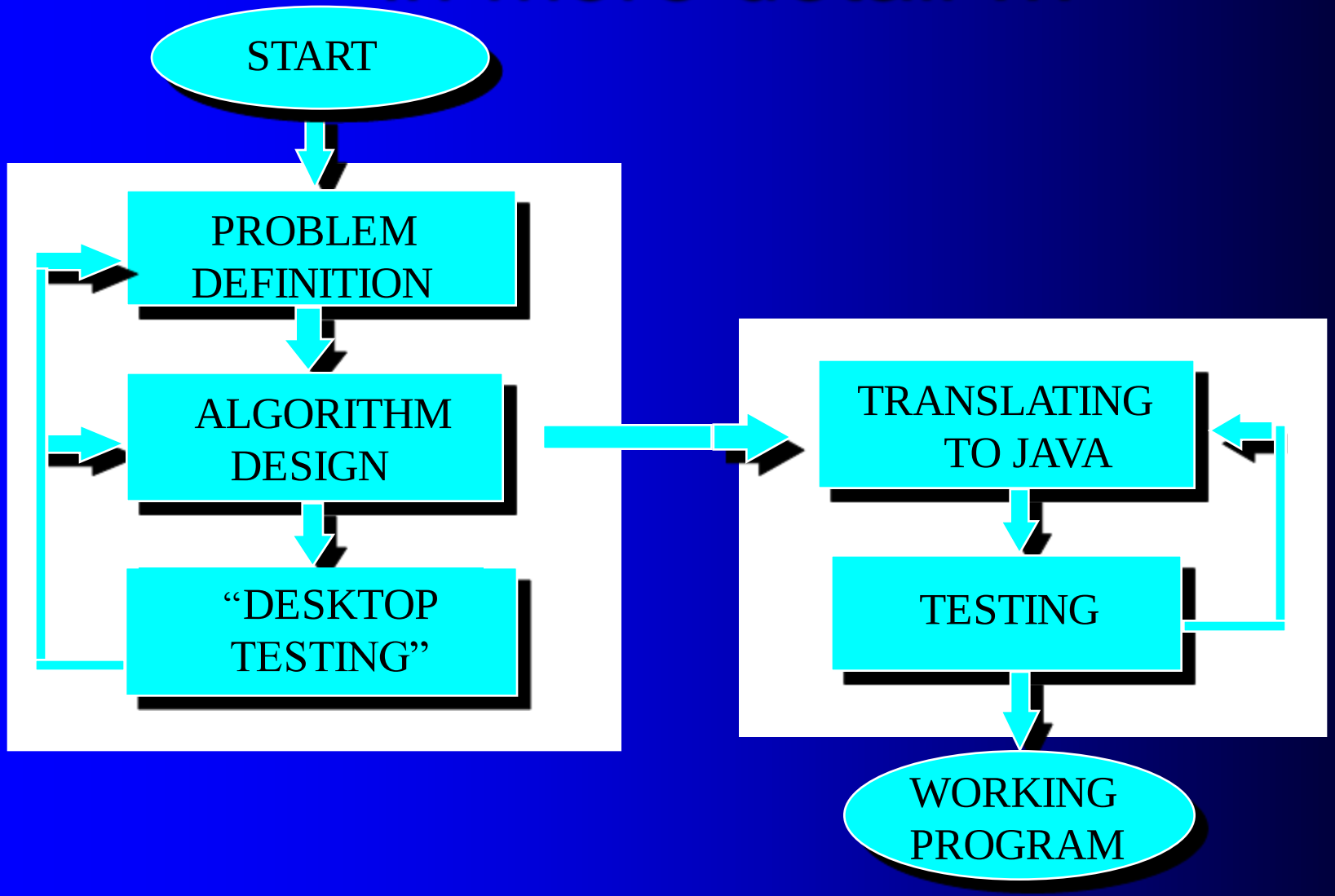
- A program is an algorithm written in some programming language
- An algorithm* is a set of instructions to solve a problem

* *To be more precise, an algorithm is a finite sequence of instructions which, when executed, solves the problem at hand*

Steps involved in solving problems on a computer

- Understand the problem
- Design the solution
- Implement (program) the solution
- Test the solution

In more detail …



Example

- Problem
  - We are planning a restaurant booking for a party. We need to know how many tables to book, given the number of guests attending the party and the number of seats at each table.

Step 1: Understand the problem

- One effective way is to think about the input and output, and perhaps solve the problem for various scenarios

Step 2: Design the algorithm

- Get the number of guests
- Get the number of seats per table
- Determine the number of tables needed
- Output the number of tables

Refine the algorithm

- Get number of guests (numberOfGuests)
- Get number of seats per table (tableSize)
- Calculate numberOfTables to be the least integer greater than or equal to the division

  numberOfGuests / tableSize
- Output numberOfTables

Step 3: Convert to a Java program

- Start with basic program

```
public class Party
{
    public static void main(String[ ] args)
    {
        // Get number of guests
        // Get number of seats per table
        // Calculate number of tables needed
        // Output number of tables
    }
}
```

Be careful, Java is case-sensitive

- Add instructions

```
import java.util.*;
public class Party
{
    public static void main(String[ ] args)
    {
        // Get number of  guests
        Scanner keyboard = new Scanner(System.in);
        System.out.print("Please enter the number of guests: ");
        int numberOfGuests = keyboard.nextInt();

        // Get number of seats per table
        System.out.print("Please enter the number of seats per table: ");
        int tableSize = keyboard.nextInt();

        // Calculate number of tables needed
        int numberOfTables =
         (int) Math.ceil( (double) numberOfGuests /  tableSize );

        // Output number of tables
        System.out.println("Then you will need " + numberOfTables + " tables.");
    }
}
```

Step 4: Create and test the program

- Use vi to edit the program

  > vi Party.java

- Compile the program

  > javac Party.java

- Run the program

  > java Party

Results of tests

Please enter the number of guests: 23
Please enter the number of seats per table: 4
Then you will need 6 tables.

Please enter the number of guests: 12
Please enter the number of seats per table: 6
Then you will need 2 tables.

Please enter the number of guests: 0
Please enter the number of seats per table: 3
Then you will need 0 tables.

Please enter the number of guests: -1
Please enter the number of seats per table: 2
Then you will need 0 tables.

Testing and debugging

- Bug: an error in a program
- Debugging: the process of finding and removing bugs

picture from www.history.navy.mil/photos/pers-us/uspers-h/g-hoppr.htm

Testing and debugging

- Compiling: before the compiler converts a program into byte code or object code it must first check that the source code is correct
- Just because a program compiles does not mean it is bug free
- It must be thoroughly tested for less obvious error such as errors in logic

Errors in programs

- Compile-time error
  - Lexical error such as an invalid identifier name
  - Syntax error which is a mistake in the form of the program such as a missing semicolon
  - Semantic error which is a mistake in the meaning of a program such as not declaring a variable before it is used
  - All reported by the compiler

Errors in programs

- Run-time error
  - Occurs during execution and causes the program to stop
- Logical error
  - The program compiles and executes but produces the wrong answer

Types

- The concept of type applies to both simple data and objects
- The type of a simple value can be byte, short, int, long, float, double, boolean, or char
- The type of an object is the class to which it belongs
  - The terms object and instance are normally used interchangeably

Next lecture

- Algorithms in detail
  - Sequence
  - Selection
  - Repetition