ANDIJAN STATE PEDAGOGICAL INSTITUTE

Lecture materials · 2026–2027 academic year

SUBJECT

Information Systems and Digital Technologies

Lecture 13 · Application Software and Software Development Lifecycle

Part 2 of 2 · 2 hours

Lecturer: **Ismailov Alisher**

E-mail: alisherismailov1991@gmail.com

Contact: **99-499-73-64**

LECTURE 13 · 2 HOURS · PART 2/2

FUNDAMENTALS OF COMPUTERS · CHAPTER 10

Computer Software

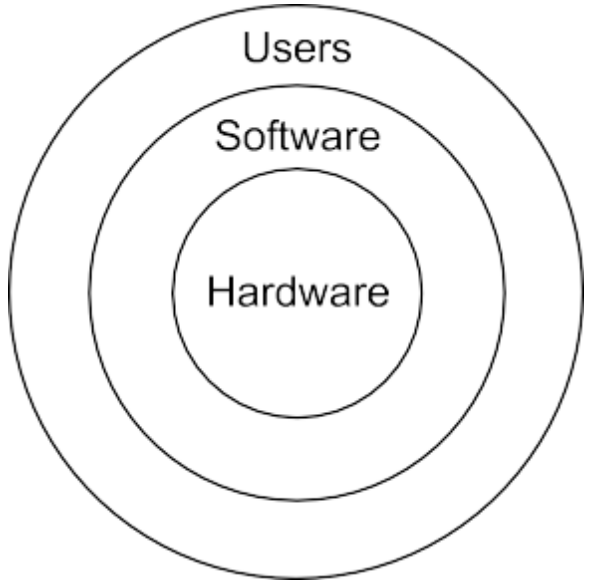
The programs that give the hardware life — and how we plan them

System management

System development

Standard applications

Unique applications



Software sits between users and hardware — textbook Fig. 10.1

By the end of this lesson, you can…

4	Plan a solution with a hierarchy chart, algorithm, flowchart and pseudocode.
5	Identify sequence, selection and repetition structures in a program.
7	Describe the phases of the software development life cycle (SDLC).
8	Compare the waterfall and incremental models and choose one for a project.

No common sense, so we plan every step

1 Hierarchy chart

Break the big problem into small tasks (top-down).

→

2 Algorithm

List the steps in English phrases.

→

3 Flowchart

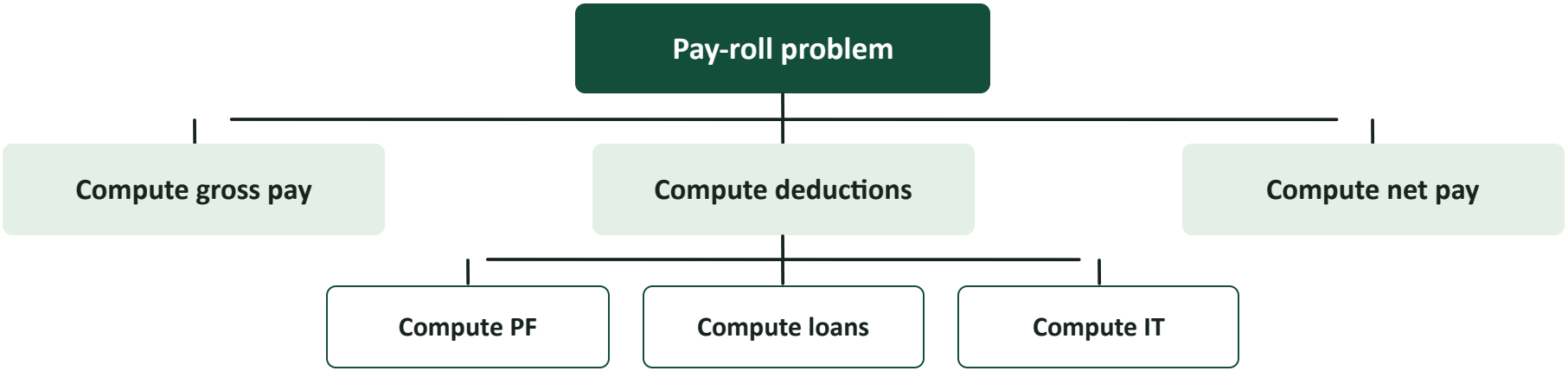
Draw the steps as a picture.

→

4 Pseudocode

Write the logic in generic, code-like statements.

EXAMPLE: HIERARCHY CHART FOR A PAYROLL PROBLEM



Redrawn from textbook Fig. 10.13

Same problem, two ways: add two numbers

ALGORITHM (ENGLISH STEPS)

Step 1 Accept the first integer from the user (integer1).

Step 2 Accept the second integer (integer2).

Step 3 Calculate integer3 = integer1 + integer2.

Step 4 Display integer3 as the result.

PSEUDOCODE (TEXTBOOK)

```
Define: Integer num1, num2, result.
Input: Integer num1.
Input: Integer num2.
Sum: result = num1 + num2
Output: Display(result).
```

Which algorithm is best? The one that needs less time (time complexity) and less memory (space complexity).

Flowcharts: six symbols



Start / end
oval



Input / output
parallelogram



Action / process
rectangle



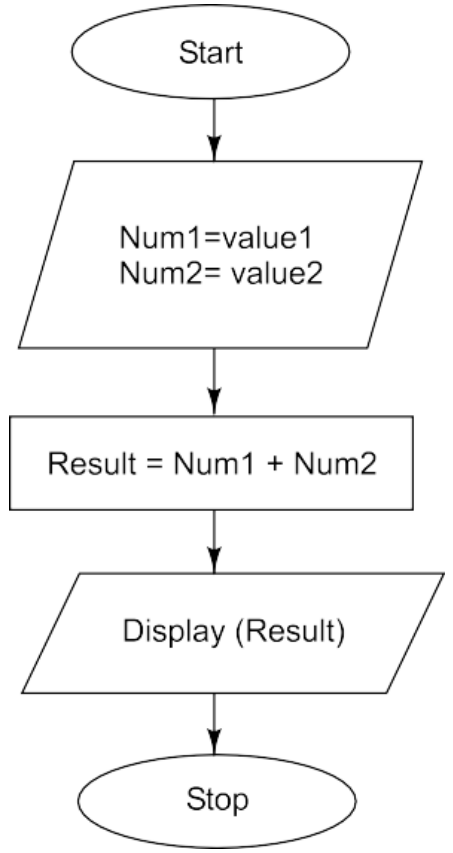
Decision
diamond



Flow of steps
arrow



Connector
circle

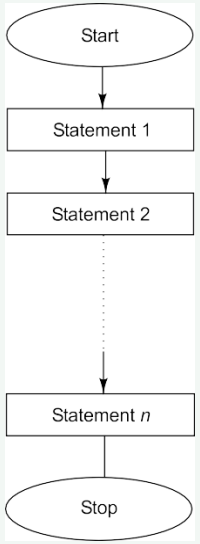


Adding two numbers — textbook Fig. 10.20

Three control structures

Sequence

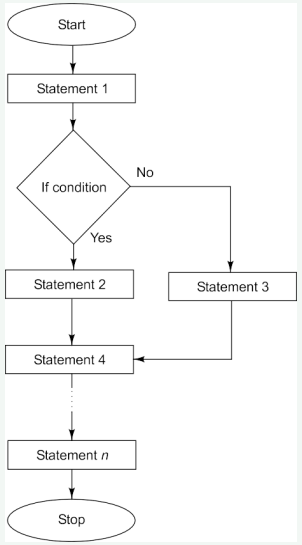
Statements run in order, one after another. No conditions.



Textbook Fig. 10.21

Selection (branching)

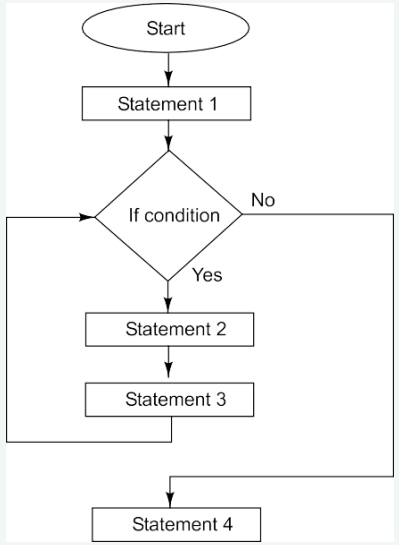
A condition decides which block runs: If-Then, If-Then-Else, Case.



If-Then-Else — Fig. 10.22

Repetition (looping)

A block repeats while a condition holds: Do-while, Repeat-until.



Do-while — Fig. 10.23

CHECK FOR UNDERSTANDING

Trace the logic

A

```
marks = 42
IF marks >= 50 THEN
    Display "Pass"
ELSE
    Display "Fail"
```

What is displayed? Which structure is it?

B

```
x = 1
DO
    Display x
    x = x + 1
WHILE x <= 3
```

What is displayed? Which structure is it?

C Which flowchart symbol would you draw for: **"Is age ≥ 18?"** and for **"Display the result"** ?

Solving a problem with a computer: three phases

1 Understand the problem

- Identify parameters and constraints
- Collect information (old reports, documents)

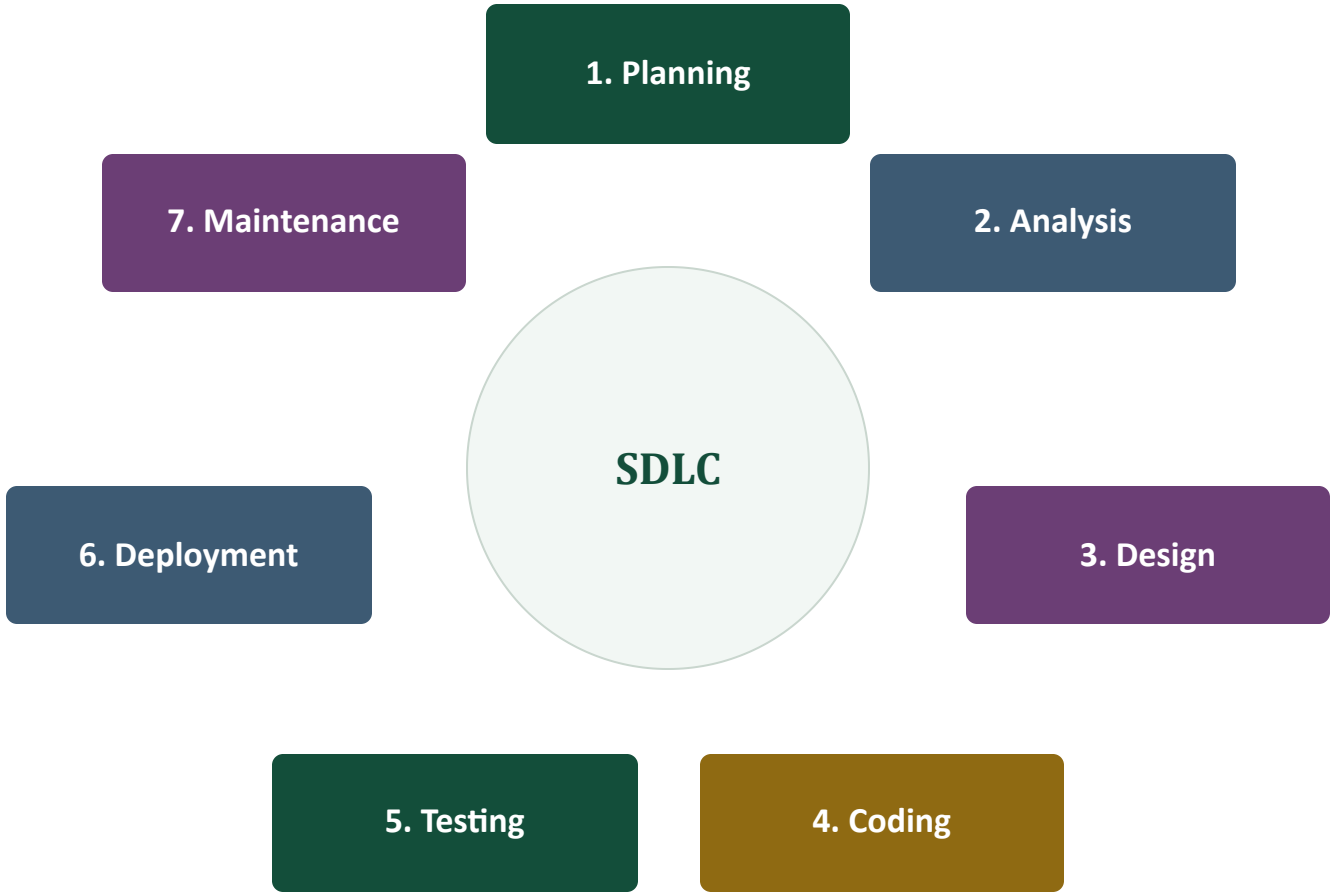
2 Develop a program

- Plan the logic: algorithm, flowchart, pseudocode
- Write the code — correct syntax and meaning
- Debug to remove errors

3 Execute the program

- Run the error-free program
- See the solution

The software development life cycle (SDLC)



What happens in each phase

1 Goal, budget, time plan

2 Collect user requirements

3 Plan screens, data, structure

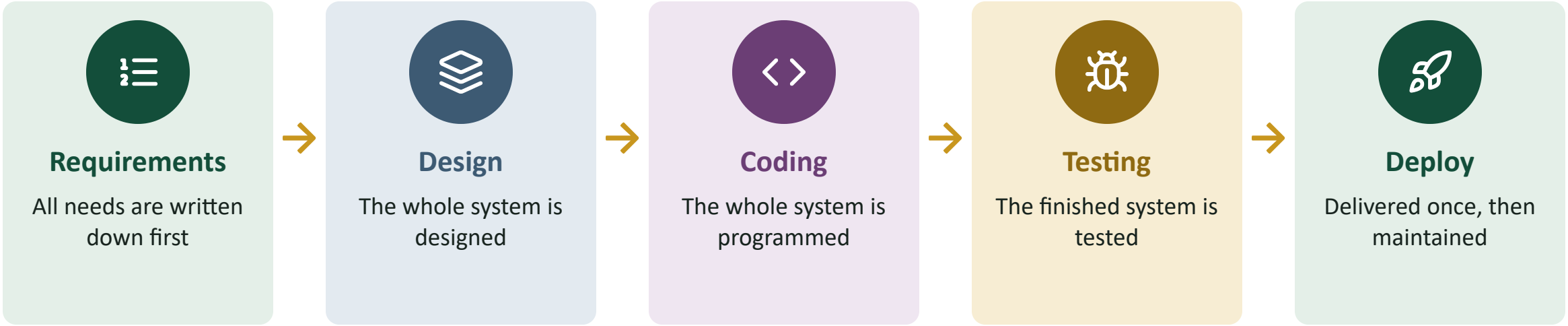
4 Write the program

5 Find and fix bugs

6 Install and train users

7 Fix, update, improve

The waterfall model: one phase after another



Strengths

Simple, clear order; good documents.

Works when requirements are fixed and well known.

Weaknesses

Hard to go back and change earlier phases.

Users see the product only at the end.

The incremental model: build it piece by piece



Example: a messaging app

v1: sign up and text chat → v2: photos and groups → v3: voice and video calls.

Strengths and weaknesses

+ Working software early; easy to change; lower risk.

– Needs good overall design; many releases to manage.

Waterfall vs incremental

	Waterfall	Incremental
Order of work	Each phase once, in a line	Repeated mini-cycles, one per increment
Users see software	Only at the end	After the first increment
Changing requirements	Difficult and costly	Easier — in the next increment
Testing	Late, on the whole system	Early and often, piece by piece
Best for	Fixed, well-known needs (e.g. medical device)	Changing needs (e.g. mobile app, website)

Think: are the requirements fixed, or will users change their minds?

CHECK FOR UNDERSTANDING

Life cycle and models

01 The team interviews shop staff to learn what the new system must do.

02 Testers find that the “Pay” button crashes the app.

03 Two years after launch, the app is updated for a new Android version.

04 Screens and the database are drawn before any code is written.

05 Software for a pacemaker, with strict, fixed rules.

06 A start-up releases a simple app now and adds features every month.

Choose from:

Analysis	Design	Testing	Maintenance	Waterfall	Incremental

Software around you

TEXTBOOK



Antivirus running in the background

→ **Utility program**

TEXTBOOK



Installing a driver for a new printer

→ **Device driver**

TEXTBOOK



A code editor for your programming class

→ **Source code editor**

TEXTBOOK



Class marks in a spreadsheet

→ **Standard application**

TODAY



Supermarket checkout scanning barcodes

→ **Inventory management**

TODAY



Apps you install on a phone

→ **Application software**

Review: can you answer these?

01 What is the difference between system and application software?

02 Name three utility programs and what they do.

03 Why does a computer need device drivers?

04 How is a compiler different from an interpreter?

05 What does a linker do?

06 What do file, record and field mean in a DBMS?

07 Name the four steps of problem solving.

08 How is Repeat-until different from Do-while?

Exit ticket

Answer on a slip of paper before you leave.

1. Sort into system or application software: MS Excel, device driver, compiler.

2. Which flowchart symbol shows a decision, and which shows input or output?

3. Write a 3-step algorithm to find the area of a rectangle.