ES 1036 Programming Fundamentals

Unit 1: Basic Terminology and problem solving steps

Dr. Quazi M. Rahman , Ph.D, P.Eng, SMIEEE
Office Location: TEB 361
Email: QRAHMAN@eng.uwo.ca
Phone: 519-661-2111 x81399

"There are no secrets to success. It is the result of preparation, hard work, and learning from failure" ~Colin Powell

"Genius is 1% talent and 99% percent hard work..." ~ Albert Einstein

Outline

- **Basic terminology for computers**
- **Problem-solving methodology for engineering problems**

Definitions

Computer:

- Programmable machine designed to follow/process instructions at an enormous speed

Computer Program (Also known as software):

- A list of instructions directing the computer to perform a task

Note: Programming is a way of thinking, not a rote skill

Definitions, *cont*

Hardware:

Ø Refers to the computer equipment, e.g., keyboard, mouse, terminal, hard disk, printer

Software:

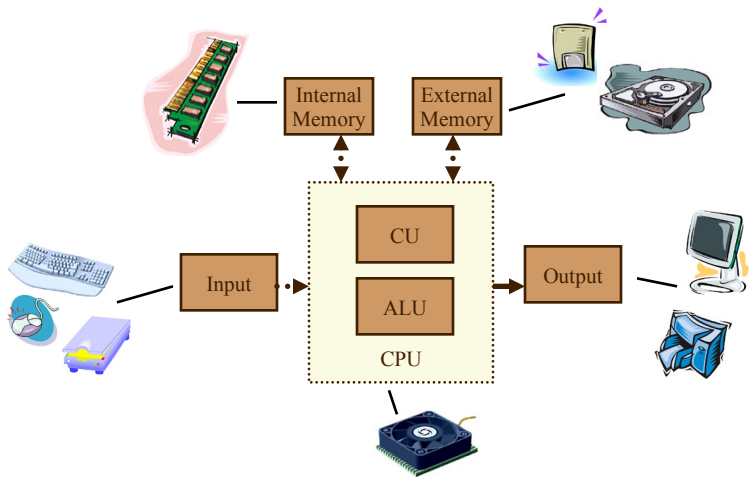
Ø Refers to the computer programs

Ø Example: An operating system is a software that provides a convenient and efficient interface between the user and the hardware

Computer system:

Ø The combination of hardware and software

Computer Hardware



Computer Hardware, *cont.*

CPU (Central Processing Unit):

- Control Unit (**CU**) : Controls the computer resources, also called **Processor**
- Arithmetic and Logic Unit (**ALU**): Performs mathematical operations
- Note: CPU has some memory which are called the registers and present-day **CPU** uses the cache **memory** too. BUT we **do not** consider this memory as part of the internal memory.
- Cache memory: An auxiliary memory, generally inaccessible, from which high-speed retrieval is possible

Computer Hardware, *cont.*

Input Devices:

- Devices used to input information in to the CPU of the computer
- Example:
  - keyboard, mouse, scanner etc.

iClicker Attendance??????

Computer Hardware, *cont.*

Output Devices :

- Devices used to make the computer-processed data/information available at the output
- Examples:
  - Computer monitor, printer.

Computer Hardware, *cont.*

Internal Memory:

- Holds both program instructions and data
- When any data is inputted through the input device, it directly goes to the memory
- Also called primary storage and composed of:
  - Random Access Memory (**RAM**)
    - Volatile – requires constant power to maintain the stored information, erased when computer is turned off
  - Read-Only Memory (**ROM**) (Volatile or NON volatile?)

Computer Hardware, *cont.*

External Memory :

- Non-volatile - data retained even if the computer is turned off
- It is suitable for long-term storage of information, also called secondary storage
- Examples: floppy disk, hard drive, flash memory, CD, Solid State Device (SSD)

iClicker Attendance??????

Review

1) A CPU consists of an ALU, scanner, and a processor

   a) True

   ✓ b) False

Review

2) Instructions and data are stored in

   a) the arithmetic logic unit (ALU)

   b) the control unit (processor)

   c) the central processing unit (CPU)

   ✓ d) the memory

   e) the keyboard

Review

3) An operating system is

a) The software that is designed by users

b) A software that provides a convenient and efficient interface between the user and the hardware

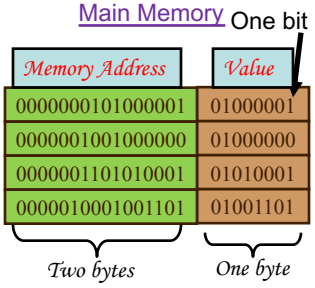
c) The set of utilities that allow us to perform common operation

d) A set of software tools

iClicker Attendance??????

Main Memory Organization

- Main memory is divided into numbered locations each of which contains one **byte** of data/information (in general) as value.
  - Byte is a sequence of **8 bits**
  - Bit stands for **b**inary dig**it** which can have either **0** (OFF, FALSE) or **1** (ON, TRUE) value.
- The location number associated with a byte is called the **address**



- Each location has a unique address which is generally more than one byte long.
- For any computer system the address length (in terms of bytes) is always the same
  - *No two memory locations can have the same address*

How Data is Stored?

- Computers use zeros and ones because digital devices have two stable states, which are referred to as zero and one by convention.
- The programmers need not to be concerned about the encoding and decoding of data, which is performed automatically by the system based on the encoding scheme.
- The encoding scheme varies. For example, character 'J' is represented by 01001010 in one byte.
- If computer needs to store a large number that cannot fit into a single byte, it uses more than one adjacent bytes.

Computer Languages

- Computers do not understand human languages, so we need to use computer languages to communicate with computers
- We tell a computer what to do through programs
- Programs are written using programming languages

Computer Languages, *cont.*

- **Machine Language:**
  - A computer hardware understands this language only
  - Used for communication with computer hardware directly
  - Is written using two symbols represented by **0** and **1**
  - Also called **low-level** language, or **binary language**
  - Machine (CPU/Processor) dependable
  - Example:

```
0010 0000 0000 0100
1101 1010 1001 1010
1000 0000 0000 0101
```


Computer Languages, *cont.*

- **Assembly Language:**
  - This language assigns short names to commands (e.g. ADD, MOV) instead of binary instructions as done for machine language
  - Assembly language is easier to understand when compared to machine language
  - Must be translated to machine language
  - The *assembler*, a computer program, translates the assembly language in to machine language.
  - Still machine/processor dependent
  - Example:

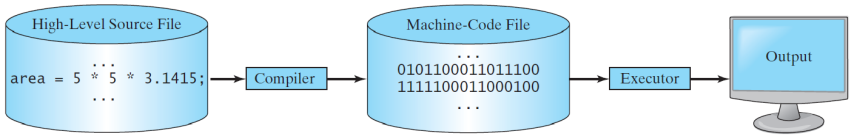
```
ADD R1, R2, R3
```

Computer Languages, *cont.*

- **High-level Language:**
  - Uses English-like command and instruction
  - Example: the following statement computes the area of a circle with radius 5: `area = 5 * 5 * 3.1415;`
  - A program written in a high-level language is called a **source program** or **source code**.
  - Easier than machine and assembly languages
  - Processor/machine independent
  - Must be translated into machine language. This translator is known as compiler.
  - Based on the process of translation, the compiler can be called as an interpreter.

Compiler

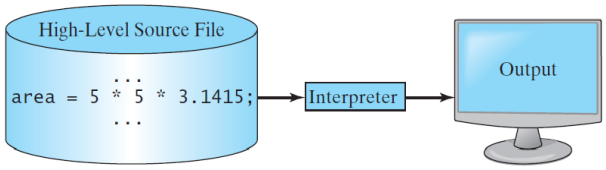
- A general compiler translates the entire source code into a machine-code file (known as object-code), and the machine-code file is then executed. C++ uses a compiler.



- *Difference between general compiler and interpreter:* https://techdifferences.com/difference-between-compiler-and-interpreter.html

Interpreter, a Special Compiler

- A special compiler called an interpreter reads one statement from the source code, translates it to the machine code, and then executes it right away.
- A statement from the source code may be translated into several machine instructions.



What Does Java use? Compiler or Interpreter

- The Java programming language does not fit into either the compiled language or interpreted language models.
- The Java compiler (javac) converts the source code into a virtual machine code called bytecode which is machine independent.
- This bytecode file (.class file) can be run on any operating system (Windows, UNIX, Mac OS) by using the Java interpreter (java) for that platform.
- The interpreter is referred to as Java Virtual Machine (JVM). Thus, Java is an example of a *Virtual Machine programming language*.
- A virtual machine (VM) is a software implementation of a machine (computer) that executes programs like a real machine.

Popular High-Level Languages

- COBOL (COmmon Business Oriented Language)
- MATLAB (MAtrix LABoratory; a numerical computing language)
- FORTRAN (FORmula TRANslation)
- BASIC (Beginner All-purpose Symbolic Instructional Code)
- Pascal (named after Blaise Pascal)
- Ada (named after Ada Lovelace)
- C language (one of the root languages)
- Visual Basic (Basic-like visual language developed by Microsoft)
- Java/C# (object oriented; developed from C and C++)
- Python (object oriented; written in C)
- C++ (an object-oriented language, modified version of C)

(FYI) For your reading pleasure browse the following site:

https://simple.wikipedia.org/wiki/Programming_language#Declarative_vs._Imperative_programming

Review

1) Assembly language is a high level language

True

False

Review

2) A high level language is CPU specific (processor/machine dependent)

True

False

Review

3) The source code in Java is compiled into Java Virtual Machine code called bytecode, which is machine-independent.

a) True

b) False

Review

4) If you change your code, you must compile it for the change to be effective

   a) True
   b) False

Review

5) _____ is the physical aspect of the computer that can be seen.

   a) Hardware
   b) Software
   c) Operating system
   d) Application program

Review

6) _____ is the brain of a computer.

a) Hardware
b) CPU
c) Memory
d) Disk

Review

7) Why do computers use zeros and ones?

a) because combinations of zeros and ones can represent any numbers and characters.
b) because digital devices (e.g., computer) have two stable states and those states can be well represented by 0 and 1.
c) because binary numbers are simplest.
d) because binary numbers are the bases upon which all other number systems are built.

Review

8) One byte has _____ bits.

a) 4
b) 8
c) 12
d) 16

Review

9) _____ translates high-level language program into machine language program.

a) An assembler
b) A compiler
c) CPU
d) The operating system

Review

10) _____ is a program that runs on a computer to manage and control a computer's activities and provides interface between the user and the computer hardware.

a) Operating system
b) C++
c) Modem
d) Interpreter
e) Compiler

Review

11) True or False: In the RAM section, any two memory locations can NOT have the same address (location number).

a) True
b) False

Review

12) Which one of the following hardware components stores the data and instructions?

a) the Arithmetic and Logic Unit (ALU)
b) the Random Access Memory (RAM)
c) the control unit (processor)
d) the Central Processing Unit (CPU)
e) the keyboard

Problem-solving methodology for engineering problems

iClicker Attendance??????

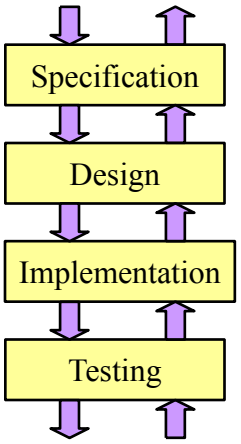
Engineering Problem Solving

- Problem Example:

Find the tripled value (times 3) of an integer number. The number should be with in range between 4 and 23 inclusive.

Engineering Problem Solving

- **Specification**
- **Design**
- **Implementation**
- **Testing**



Specification

- Developed in response to the “requirements”
- Precise statement of what the system will do
- Include
  - Functional specifications
  - Input/output specifications
  - Performance specifications
- Most important, hardest, most neglected

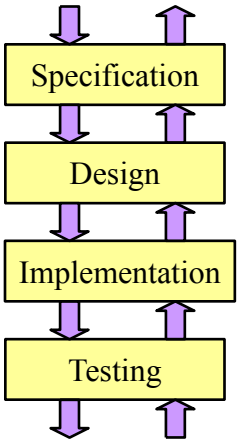
iClicker Attendance??????

Specification - example

- Requirements
  - Find the tripled value of an integer number. The number should be with in range between 4 and 23 inclusive.
- **Specifications (***One probable thought-process***)**
  1. Request user to enter any number
  2. Read the entered number
  3. Check the number: if less than 4 or greater than 23 display “Invalid” and terminate
  4. Calculate the tripled value of the number
  5. Display the result
  6. Terminate

Engineering Problem Solving

- **Specification**
- **Design**
- **Implementation**
- **Testing**

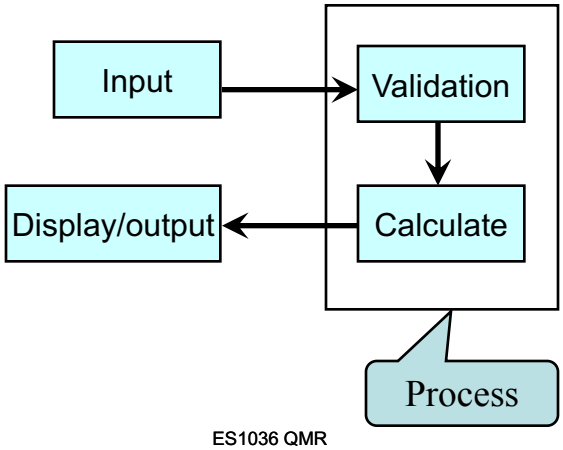


Design

- Architecture
  - Decompose the system into components
- Interfaces
  - How the components talk to each other
- Detailed design
  - How components are designed
  - Each component can be another system

Design - example

- Decompose the system into components
- How the components talk to each other

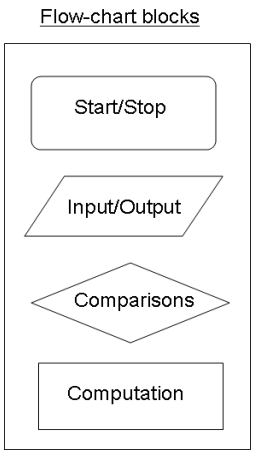


Design, *cont.*

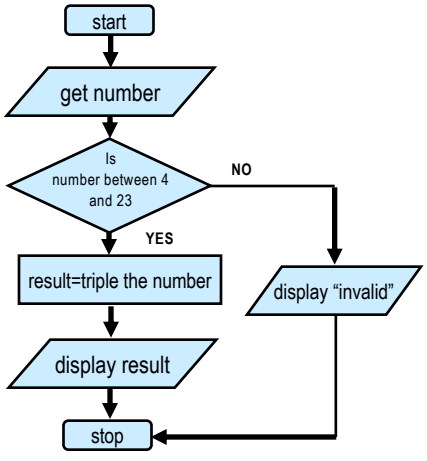
- Algorithm, (after Al Kho-war-iz-mi a 9th century Persian mathematician)
  - An ordered sequence of well-defined instructions that gives an initial state, performs some task and halts in finite time
- Algorithms can be illustrated by one of the following ways:
  - Pseudo-code
    - Kind of structured English for describing algorithms
  - Flow-Charts
    - Often used to represent algorithms graphically

Flow-Charts in Design

- Flowchart uses specific diagrams connected by arrows (→) to represent the control-flow of the algorithm



Flow-Charts in Design - example



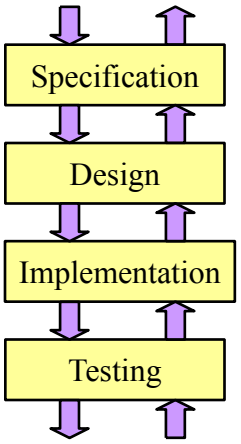
Creative Ways To Brainstorm Ideas

- *Watch out: "Never go into a **brainstorming** session without a clear **idea** of what you want out **of it**"*
  - *Specification! Specification! Specification!*

https://www.youtube.com/watch?v=yAidvTKX6xM

Engineering Problem Solving

- **Specification**
- **Design**
- **Implementation**
- **Testing**



Implementation

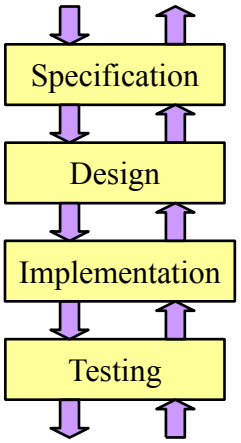
- Easiest part
- There shouldn't be any surprises
- Pick the platform
  - PC, Mac, Palm
- Pick the language
  - Java, C++, C, Ada, C#, Python
- Pick the IDE (Integrated Development Environment)
  - Eclipse, Netbeans, MS Visual Studio, Borland C++ Builder, xcode etc

Implementation - example

```
import java.util.Scanner;
public class MyClass {
    public static void main(String args[]) {
        int anyNumber=10;
        Scanner input = new Scanner(System.in);
        System.out.print("Enter any Number: ");
        anyNumber = input.nextInt();
        if (anyNumber >= 4 && anyNumber <= 23)
           System.out.println(anyNumber * 3);
         else
           System.out.println("Invalid");
    }
}
```

Engineering Problem Solving

- **Specification**
- **Design**
- **Implementation**
- **Testing**



Testing

- Second most important part of the process
- Ensures that the implementation fulfils the specification
- A test plan can be generated directly from specifications
- An item in the test plan contains details about:
  - Steps that must be followed,
  - Inputs that must be given and
  - Outputs that must be observed
- An independent team of engineers not involved in the design and implementation of the project usually conducts such testing.

Testing - example

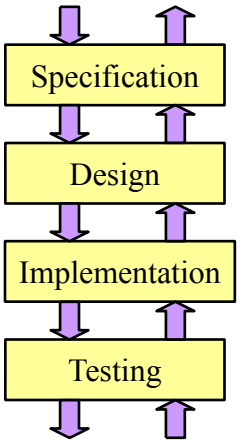
1. Start program
2. Enter '-1'. Observe "Invalid"
3. Enter '56'. Observe "Invalid".
4. Enter '16'. Observe "48".
5. Program must exit

```
Scanner input = new Scanner(System.in);
System.out.print("Enter any Number: ");
anyNumber = input.nextInt();
if (anyNumber >= 4 && anyNumber <= 23)
   System.out.println(anyNumber * 3);
else
   System.out.println("Invalid");
```

- Ask yourself: Have we considered all probable test points?
- If the answer is YES, repeat these steps number of times and watch the program outputs

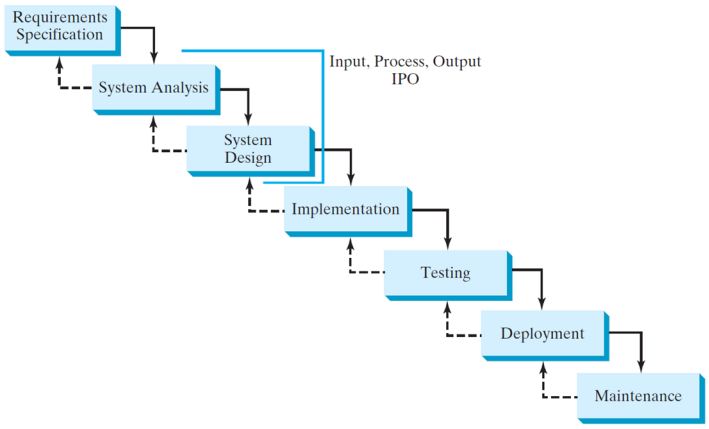
Engineering Problem Solving

- **Specification**
- **Design**
- **Implementation**
- **Testing**



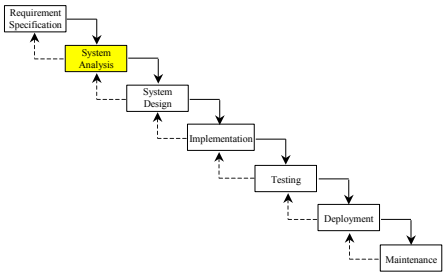
Software Development Process (SDP)

- SDP adds few more steps in between the steps we discussed on Engineering Problem solving:



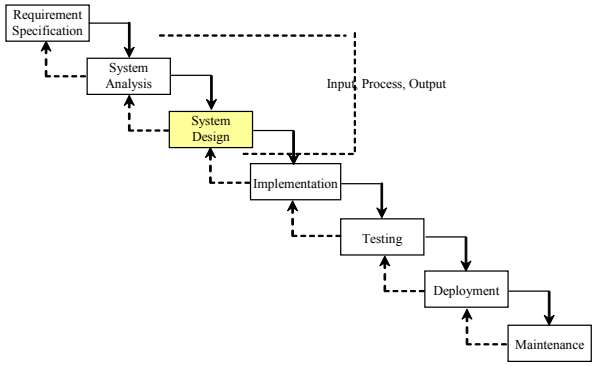
Software Development Process : System Analysis

- Seeks to analyze the business process in terms of data flow, and to identify the system's input and output.
- Part of the analysis entails modeling the system's behavior.
- The model is intended to capture the essential elements of the system and to define services to the system.



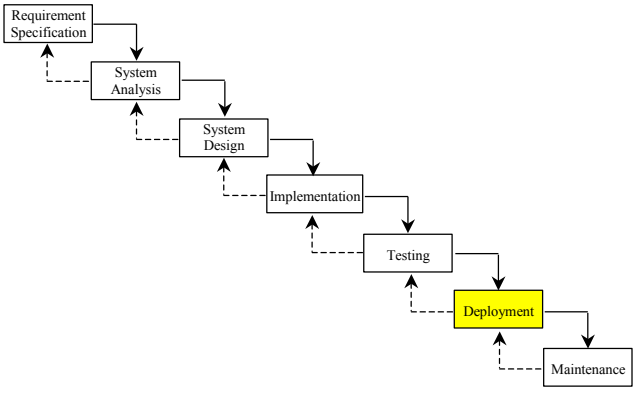
Software Development Process: IPO

- The essence of system analysis and design is input, process, and output. This is called **IPO**.



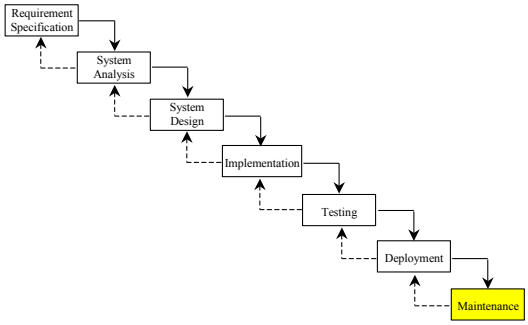
Software Development Process: Deployment

- Deployment makes the project available for use.
- For a Java program, this means installing it on a desktop or on the Web.



Software Development Process :Maintenance

- Maintenance is concerned with changing and improving the product.
- A software product must continue to perform and improve in a changing environment. This requires periodic upgrades of the product to fix newly discovered bugs and incorporate changes.



Review

13) A computer program is the implementation of an algorithm

a) True
b) False

Review

14) An algorithm refers to

a) A step-by-step solution to solve a specific problem
b) The collection of instructions that the computer can understand
c) A code that allows us to type in text materials
d) A set of math equations to derive the solution of a problem

Review

15) Writing code is done during

a) Specification
b) Design
c) Implementation
d) Testing

Review

16) In which problem solving steps, syntactical errors are corrected?

a) Specification
b) Design
c) Implementation
d) Testing

Reading Homework: Case Study

- **Problem** Your summer surveying job requires you to study some maps that give distances in kilometers and some that use miles. You and your coworkers prefer to deal in metric measurements. Write a program that performs the necessary conversion.

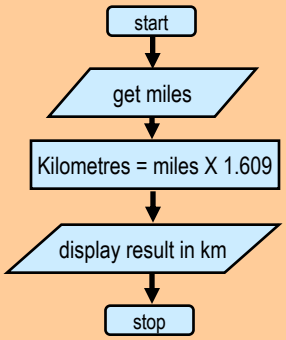
Specifications

- **Requirements:**
  - convert distance measurements (given in miles) to kilometers.
- **Specifications**
  - Problem Input
    miles *distance in miles*
  - Problem Output
    kms *the distance in kilometers*
  - Relevant Formula
    1 mile = 1.609 kilometers

Design

- Formulate the algorithm that solves the problem.
- **Algorithm**
  1. Get the distance in miles.
  2. Convert the distance to kilometers.
  3. Display the distance in kilometers.
- Algorithm Refinement
  2.1. distance in kilometers = 1.609 X the distance in miles

- Flowchart:



Implementation

```
import java.util.Scanner;
public class MyClass {
    public static void main(String args[]) {
        final double KM_PER_MILE = 1.609;
        double miles, kms;
        System.out.print("Enter the distance in miles: ");
        Scanner input = new Scanner(System.in);
        miles = input.nextDouble();
        kms = KM_PER_MILE*miles;
        System.out.println(miles+" miles is = " + kms+" kms.");
    }
}
```


Testing

- Test with input data for which one can easily determine the expected results
- E.g.
  10 miles should convert to 16.09 kilometers

Answers to the review-questions

1. B
2. B
3. A
4. A
5. A
6. B
7. B
8. B
9. B
10. A
11. A
12. B
13. A
14. A
15. C
16. C