Lecture 1: Introduction to Mobile Computing

Jianjun Chen (Jianjun.Chen@xjtlu.edu.cn)

What is Mobile Computing?

- About.com: A generic term used to refer to a variety of devices that allow people to access data and information from wherever they are.

- Wikipedia: Mobile computing is human–computer interaction by which a computer is expected to be transported during normal usage.

- UoL COMP327: The study of computing on small devices

Devices

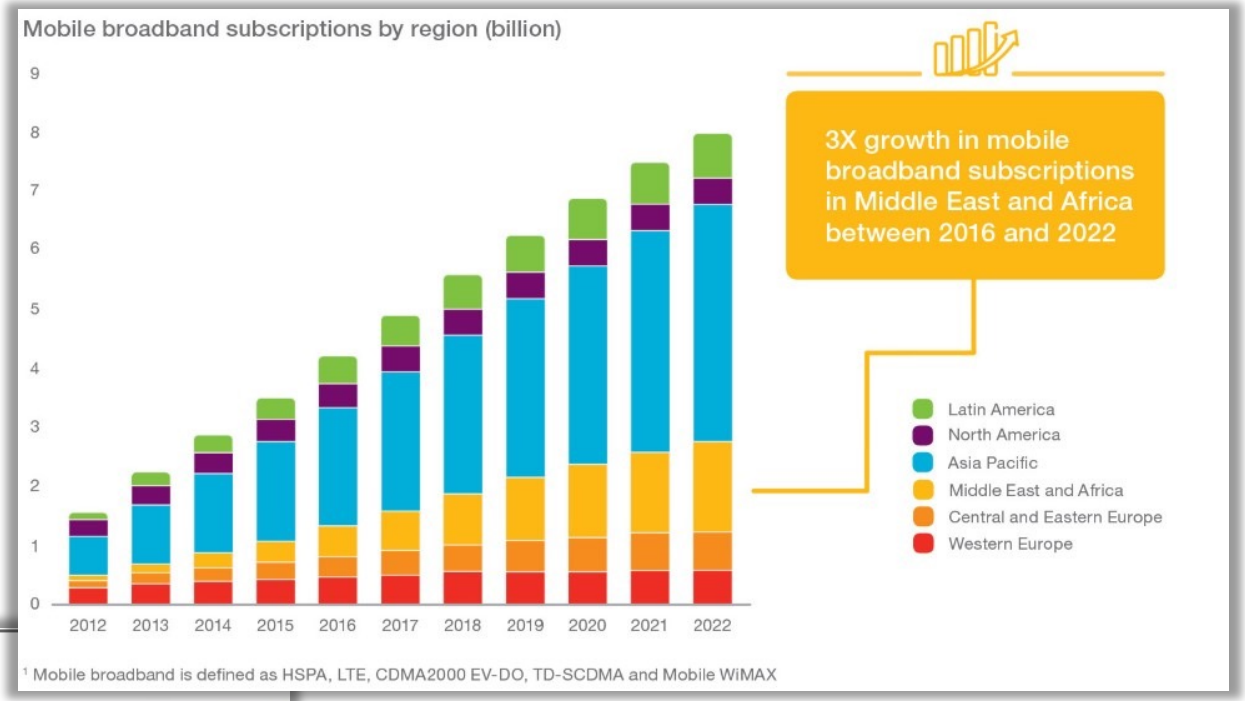
Hololens by Microsoft

Internet of things (IOT)

Applications

A mobile phone subscription refers to the use of public mobile telecommunication systems (also called mobiles or cellphones) using cellular technology.

e.g. number of active sim cards



- Over 2.2 million Android apps
- Over 2 million Apple apps
- Over 300 million downloads per year

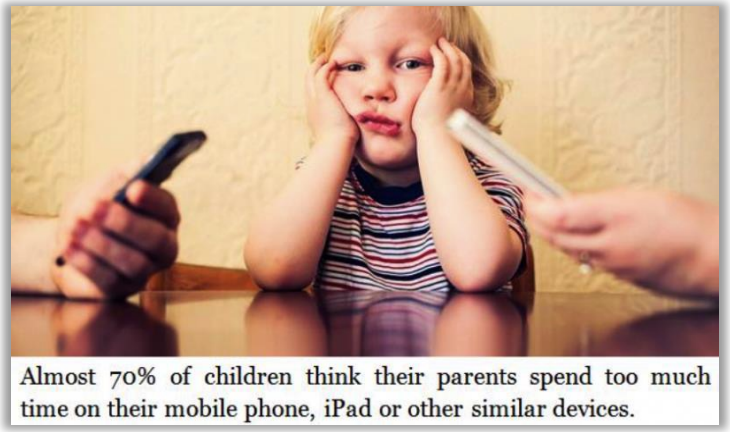
Trends & Usage

Challenges in Technologies

- Hardware - Lighter, smaller, lower energy consumption
- Software & User interface - heterogeneous devices, different interaction styles
- Network - Lower bandwidth, low reliability, higher delays, more jitter, different protocols
- Location awareness

Social Impacts

- Privacy
- Psychological Impact



Almost 70% of children think their parents spend too much time on their mobile phone, iPad or other similar devices.

Module Contents

- Review of Java programming (Week 1~2)
  - Homework
  - No lab sessions
- Android Development, using Java (Week 2~8)
  - Lab sessions starts from Week 3.
- Mobile Networking & Algorithm (Week 9~14)
  - No lab sessions
  - Mock exam paper

*I might adjust the schedule

Learning Outcomes

- Have an understanding of the characteristics and limitations of mobile hardware devices including their user-interface modalities
- Be able to develop applications that are mobile-device specific and demonstrate current practice in mobile computing contexts
- Have an appreciation of the design and development of context-aware solutions for mobile devices
- Have an awareness of professional and ethical issues, in particular those relating to security and privacy of user data and user behavior.

Learning Outcomes

- In other words, you will:
  - Know how smartphone system works.
  - Design and develop a smartphone app.
  - Address some critical issues during the design.
- This will help you in:
  - Practicing your app design and development skills.
  - Making your phone more interesting.
  - Possibly startup your own business.

Assessments

Grading Scheme:

- 85% - Individual and group android design project
- 15% - In-class project and assignment

Reference Books

- Context-Aware
  - Stefan Poslad (2009). *Ubiquitous Computing - Smart Devices, Environments and Interactions*.
  - Thomas et al. (2009). *Introduction to Algorithms (3rd edition).*
- Mobile application development
  - Official Android tutorials and manuals
  - Reto Meier (2008). *Professional Android Application Development.*
- Mobile communication
  - Martin Sauter (2006). *Communication Systems for the Mobile Information Society.*
  - Jochen Schiller (2003). *Mobile Communications (2nd edition).*

Getting Help

- Do the research yourself
  - Search engine
  - Group discussion
  - Videos online are also good.
- Office hour:
  - Jianjun Chen (Jianjun.Chen@xjtlu.edu.cn)
  - Office: SD541
  - Every Tuesday 16:00 ~ 18:00

Java Lang Review, Part I

Basic data types to compound data types

Operators to functions

Basic Types

- Limited memory space
  - 16GB on personal computer is quite common.
- Limited memory unit size (word size)
  - 32 bit registers for 32 bit CPU.
  - 64 bit registers for 64 bit CPU.
- Thus, limited quantity of numbers can be represented in one word (32/64 bits)

Basic Types

- A basic data type follows a certain way of mapping sequences of 1/0 bits to human-understandable symbols.

- For example, a real number 17:
  - Under two's-complement representation
  - Under IEEE 754 floating point representation
    - Gaps between extreme floating-point numbers.

Gaps between Floating Numbers

Actual Exponent (unbiased)	Exp (biased)	Minimum	Maximum	Gap
−1	126	0.5	≈ 0.999999940395	≈ 5.96046e-8
0	127	1	≈ 1.999999880791	≈ 1.19209e-7
1	128	2	≈ 3.999999761581	≈ 2.38419e-7
2	129	4	≈ 7.999999523163	≈ 4.76837e-7
10	137	1024	≈ 2047.999877930	≈ 1.22070e-4
11	138	2048	≈ 4095.999755859	≈ 2.44141e-4
23	150	8388608	16777215	1
24	151	16777216	33554430	2
127	254	≈ 1.70141e38	≈ 3.40282e38	≈ 2.02824e31

Basic Types

- Another example:
  - Character ‘a’ in ASCII table maps to binary number: 0110 0001
  - Same as the binary representation of decimal number 97.
- Their binary representations are identical in registers or memory.
  - However, they are treated differently by CPU instructions.
  - These instructions are generated by compilers based on both data types and operators in the source code.

Operators

- Basic “functions” provided by the programming language.
  - They behave like functions
  - “parameters” plus a “return value”.
- For example:
  - Multiplication operator (*): takes two numbers and returns another number
  - Logical negation (!): takes a Boolean and returns a Boolean.
  - Assignment operator (=): ?
    - Takes, as parameters, two expressions and returns the value assigned.

Expressions

- "An expression is a sequence of operators and operands that specifies computation of a value, or that designates an object or a function, or that generates side effects, or that performs a combination thereof."

- Computation of a value: 5 + 0.7
- A function: System.out.println()
- Side effects: y = z = 3

Question

- Is the following code an expression?

```
x = x + 3;
```

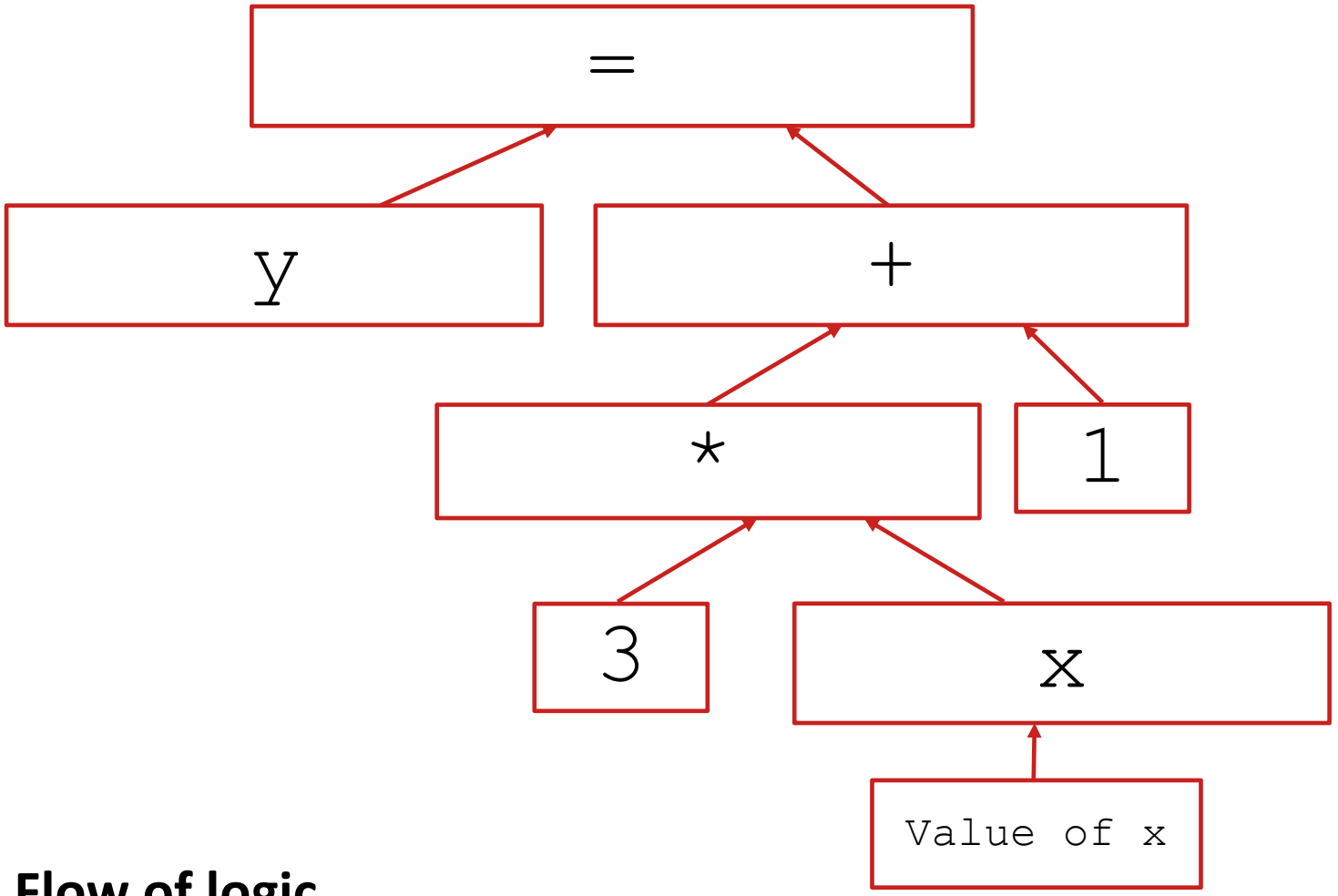
- All expressions can be evaluated into a value.
  - If/else, while, for … statements are not expressions;

Rules of Expressions

- Expression := identifier
  - `x = 5 // x is the identifier`
- Expression := constant
  - `5, 5.55, 'a'`
- Expression := string-literal
  - `"abc"`
- Expression := (expression)
- Expression := Expression + - * / = Expression
- Expression := function_name(expression, expression)

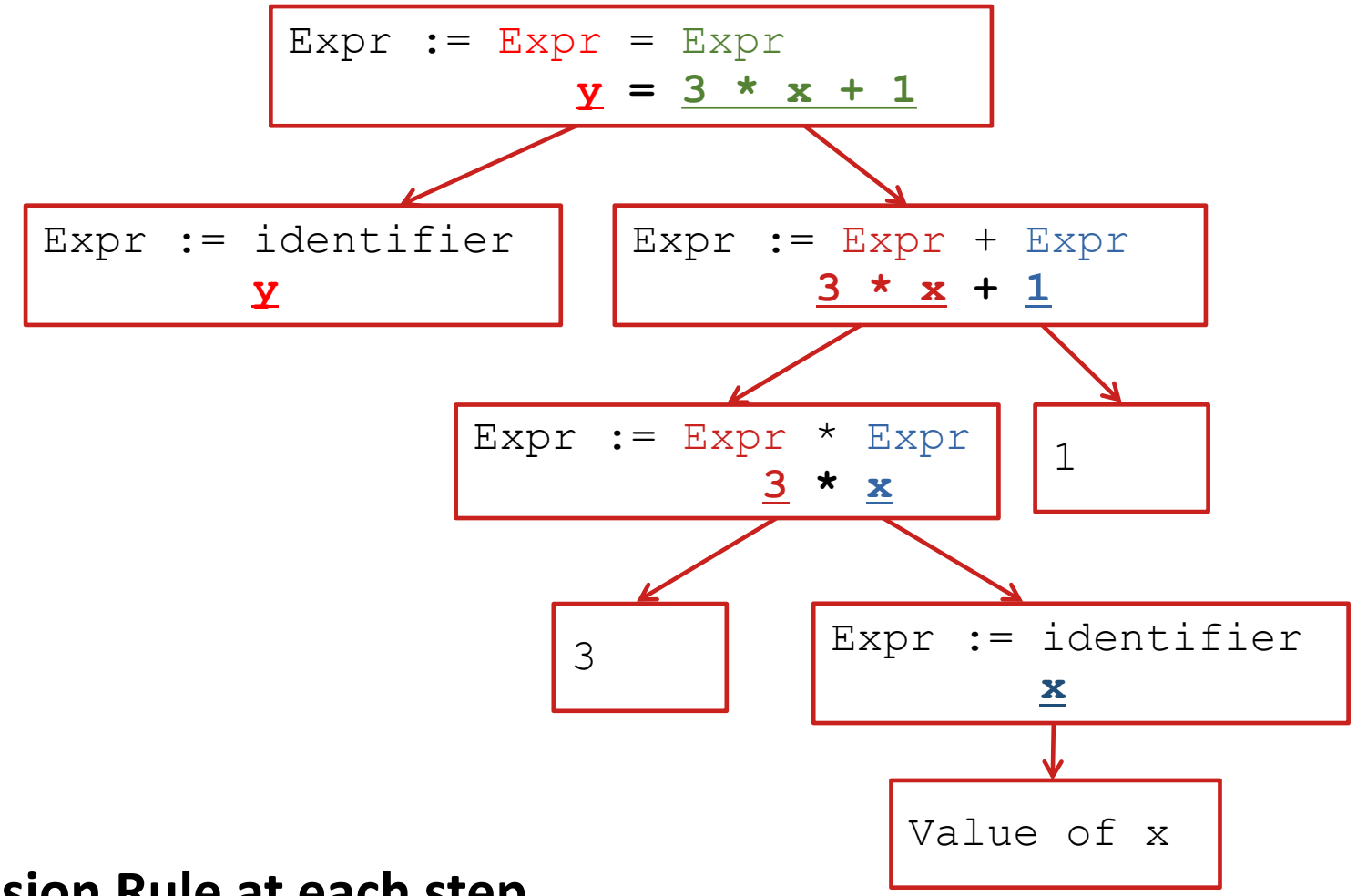
- Any expression can be interpreted as a tree. Each branch strictly follows the rules like above

An Example: y = 3 * x + 1



Flow of logic

An Example: y = 3 * x + 1



Expression Rule at each step

Expression Properties

Can you list the properties associated with the following underlined expressions?

- A[5] = 1
- a * b
- y = 12
- System.out.println()

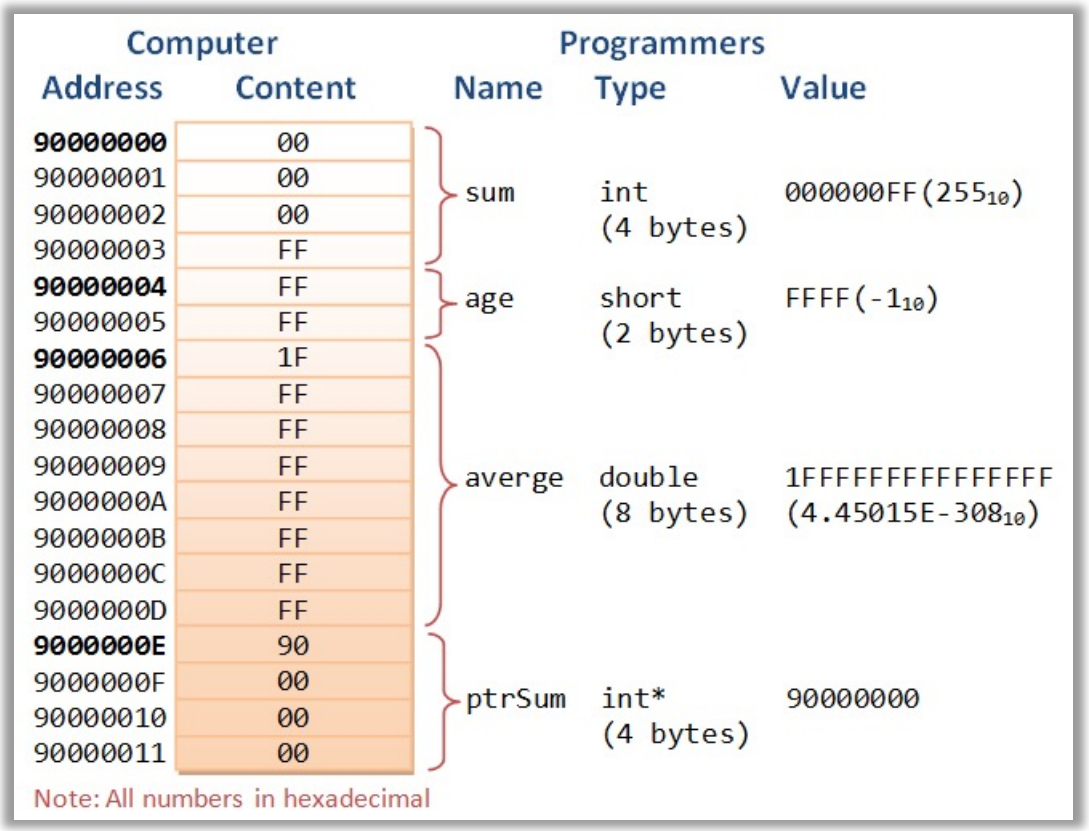
*Hint 1: what will be changed after execution? What information is used when generating instructions for them?

*Hint 2: You may start from thinking about the properties associated with a simple variable first.

Expression Properties

Three most important properties:

- Memory address
- Content
- Data type



Computer		Programmers		
Address	Content	Name	Type	Value
90000000	00			
90000001	00	sum	int (4 bytes)	000000FF(255_{10})
90000002	00			
90000003	FF			
90000004	FF	age	short (2 bytes)	FFFF(-1_{10})
90000005	FF			
90000006	1F	averge	double (8 bytes)	1FFFFFFFFFFFFFF ($4.45015E-308_{10}$)
90000007	FF			
90000008	FF			
90000009	FF			
9000000A	FF			
9000000B	FF			
9000000C	FF			
9000000D	FF			
9000000E	90	ptrSum	int* (4 bytes)	90000000
9000000F	00			
90000010	00			
90000011	00			

Note: All numbers in hexadecimal

Test 1

- Which properties are involved in the following expressions (red part)?
  - strArray[5] = “new string”
  - y = strArray[1]
- Compare the two functions on the right
  - do they consume different amount of memory?
  - How about computational time?

```
public int f1() {
    int x = f3();
    return x;
}

public int f2() {
    return f3();
}
```

Test 2

- Compare the expressions below. Assume x is 10 and y is 12:
  - `System.out.println("result is " + (x + y));`
  - `System.out.println("result is " + x + y);`

- What does the output look like? Why? Explain using the knowledge of operators.

Test 3

- When we create **objects**, what information is associated with their identifiers?

```
String str = new String();
```

- What kind of content is stored in `str`?

- Identifiers of objects store the references.
  - Reference: the memory addresses that stores data.
- The content of objects are stored in heap.
  - Keyword "`new`" allocates objects at heap.

Homework

- Can you implement for loops using only functions?

```
forLoop(???, ???, ???);
```

  - You are allowed to add helper classes.
  - `forLoop()` function must not contain `for`, `while`, `do while` statements.
  - Consider a simple case of getting the sum of the numbers in an array