OBJECT-ORIENTED PROGRAMMING TECHNOLOGY

Lecturer：陈笑沙

Material：GitHub

COURSE ARRANGEMENT

Class Schedule

24 periods of teaching contents

8 periods of experiment contents

Grade Composition

72% Exam Score, 28% Regular Performance

Regular Performance: Participation and Homework

CLASS MATERIAL

https://github.com/pdcxs/cpp-class-material

LESSON 1

1.1 Software and Hardware

1.2 Introduction of Programming Tools

1.3 Setup Development Environment

1.4 Programming Languages and Their History

1.5 The Simplest C++ Program

1.1 SOFTWARE AND HARDWARE

HARDWARE

- Input Hardware：
  1. Keyboard
  2. Mouse
  3. Camera
  4. ……
- Output Hardware:
  1. Display
  2. Printer
  3. ……

1.1 SOFTWARE AND HARDWARE

HARDWARE

- Computer Architecture:
  1. RAM (Random Access Memory)
  2. Hard Disk
  3. CPU (Central Processing Unit)
  4. GPU (Graphics Processing Unit)
  5. ……

1.1 SOFTWARE AND HARDWARE

SOFTWARE

- Operating Systems：
  1. Windows
  2. Linux
  3. Mac
- Programming：
  1. Compiler
  2. Interpreter
  3. Editor
  4. IDE (Integrated Development Environment)

1.1 SOFTWARE AND HARDWARE

SOFTWARE

Others

- Real-time communication
- Official
- Entertainment
- ...

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Compiled Language

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Compiled Language
- Interpreted Language

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Compiled Language
- Interpreted Language
- Static Typed Language

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Compiled Language
- Interpreted Language
- Static Typed Language
- Dynamic Typed Language

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Compiled Language
- Interpreted Language
- Static Typed Language
- Dynamic Typed Language
- Strong Typed Language

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Compiled Language
- Interpreted Language
- Static Typed Language
- Dynamic Typed Language
- Strong Typed Language
- Weak Typed Language

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Procedural Programming Languages
- Object-Oriented Programming Languages
- Functional Programming Languages
- Declarative Programming Languages
- Markup Programming Languages

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Procedural Programming Languages
- Object-Oriented Programming Languages
- Functional Programming Languages
- Declarative Programming Languages
- Markup Programming Languages

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Procedural Programming Languages
- Object-Oriented Programming Languages
- Functional Programming Languages
- Declarative Programming Languages
- Markup Programming Languages

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Procedural Programming Languages
- Object-Oriented Programming Languages
- Functional Programming Languages
- Declarative Programming Languages
- Markup Programming Languages

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Procedural Programming Languages
- Object-Oriented Programming Languages
- Functional Programming Languages
- Declarative Programming Languages
- Markup Programming Languages

1.2 INTRODUCTION OF PROGRAMMING TOOLS

CLASSIFICATION OF PROGRAMMING LANGUAGES

- Procedural Programming Languages
- Object-Oriented Programming Languages
- Functional Programming Languages
- Declarative Programming Languages
- Markup Programming Languages

1.2 INTRODUCTION OF PROGRAMMING TOOLS

C++

C++ is a static typed language

C++ is a weak typed language

C++ is a compiled language

C++ is a multi-paradigm language

1.3 SETUP DEVELOPMENT ENVIRONMENT

Video Tutorial (Chinese):

Setup Development Environment：
https://www.bilibili.com/video/BV1beNfepExE

1.3 SETUP DEVELOPMENT ENVIRONMENT

1. Use `scoop` To Manage Software

Setup Environment Variables

Add Bucket

Install vscode、mingw、cmake、xmake

1.2 SETUP DEVELOPMENT ENVIRONMENT

2. Configure VSCode

Install `C/C++ Plugin`

Install Xmake Plugin

1.3 SETUP DEVELOPMENT ENVIRONMENT

Conventional Compiling Process

Add Third-Part Library Manually

g++ -c source.cpp

g++ source.o

Write Makefile

1.3 SETUP DEVELOPMENT ENVIRONMENT

Mordern Compiling Process

cmake

xmake

……

1.3 SETUP DEVELOPMENT ENVIRONMENT

Homework

Setup C++ Development Environment on Any Platform, Compile and Run HelloWorld Program.

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

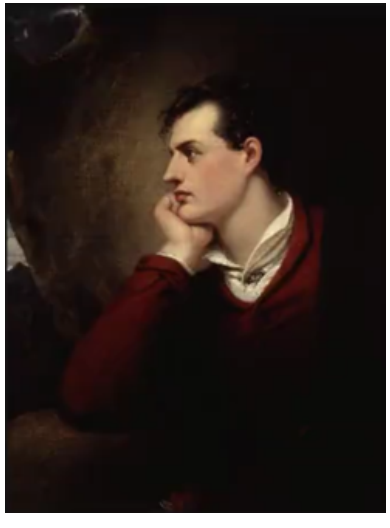
Father of C++
Bjarne Stroustrup

Father of Java
James Gosling

Father of Python
Guido van Rossum

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

THE OLD DAYS: MACHINE LANGUAGE



British Poet: George Gordon Byron

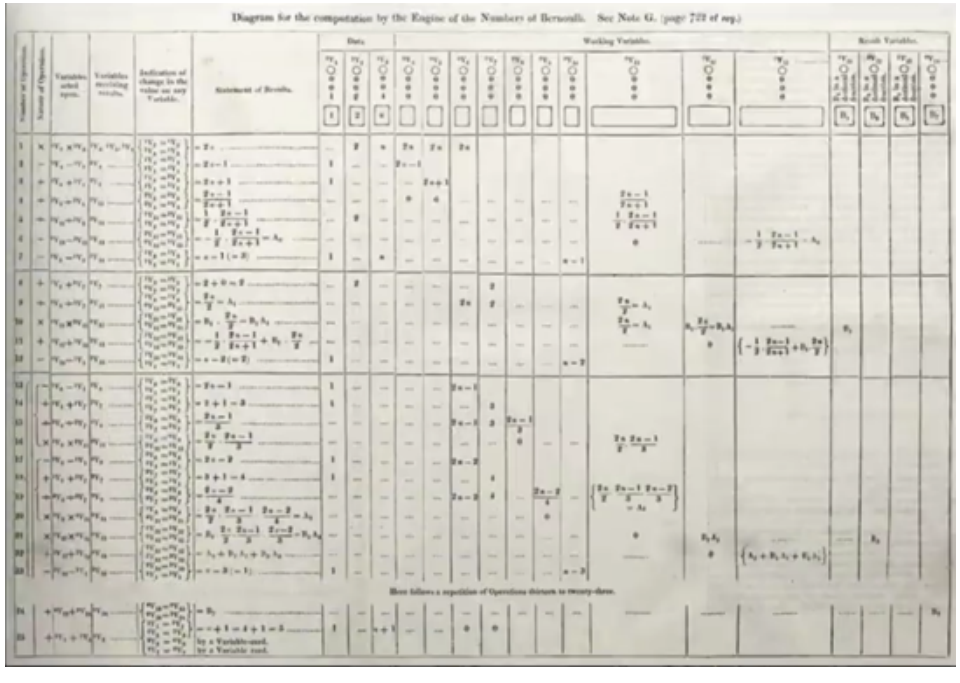
Augusta Ada King

Charles Babbage

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

THE OLD DAYS: MACHINE LANGUAGE

The Difference Engine



First Program in the World

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1930S

Haskell Curry
American
mathematician
Combinational
Logic

Alonzo Church
Lambda Calculus

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

Alan Mathison Turing

1937 Turing Machine
1939 Deciphering German Military Codes
Movie: The Imitation Game

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

John von Neumann
1945 Proposed the von Neumann architecture

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

Kathleen Hylda Valerie Booth 1947 Assembly Language

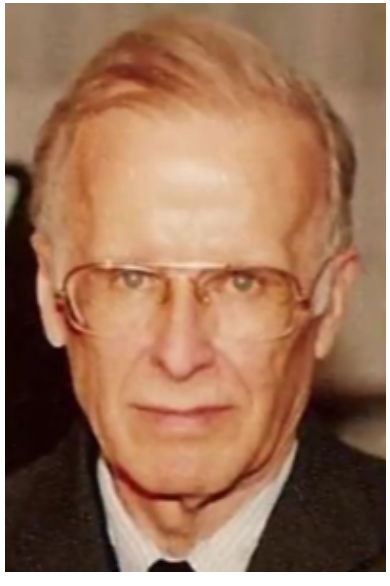
Copyright © The McGraw-Hill Companies, Inc. Permission required for reproduction or display.

Machine Language vs. Assembly Language

Objective: Multiply the value stored in R4 by 120

Machine Language	Assembly Language	
0011 0000 0000 0000	.ORIG x3000	
0101 000 000 1 00000	AND R0, R0, #0	
0101 111 111 1 00000	AND R7, R7, #0	
0001 000 000 1 ?????	ADD R0, R0, ???	
0000 110 000000011	TEST BRnz DONE	
0001 111 111 000 100	ADD R7, R7, R4	
0001 000 000 1 11111	ADD R0, R0, #-1	
0000 111 111111100	BRnzp TEST	
1111 0000 0010 0101	DONE HALT	
	.END	7-4

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY



John Warner
Backus
IBM

```
PROGRAM TRIVIAL
    INTEGER I
  I = 2
  IF(I .GE. 2) CALL PRINTIT
  STOP
END
SUBROUTINE PRINTIT
    PRINT *, 'Hola Mundo'
  RETURN
END
```

Fortran Language
1966 Fortran66
Latest: Fortran 2023

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

John McCarthy
IBM

```
(defun all-pairs (M N)
  (if (null M) nil
      (append (distl (car M) N)
              (all-pairs (cdr M) N))))

(defun distl (x N)
  (if (null N) nil
      (cons (list x (car N))
    (distl x (cdr N)))))
```

Lisp Language

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1. 1958
2. Association for Computing Machinery (ACM)
3. USA German Society for Applied Mathematics and Mechanics (GMA)
4. ALGOL58
5. In ALGOL60, concepts such as recursion, scope, and code blocks were introduced.

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

```
BEGIN
FILE F(KIND=REMOTE);
  EBCDIC ARRAY E[0:11];
  REPLACE E BY "HELLO WORLD!";
  WRITE(F, *, E);
END.
```

```
begin
    printf(($gl$, "Hello, World!"))
end
```

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1958

Grace Murray Hopper
IBM

```
IDENTIFICATION DIVISION.
PROGRAM-ID. MAIN.

DATA DIVISION.
  WORKING-STORAGE SECTION.
  01 WS-STUDENT-ID PIC 9(4) VALUE 1000.
  01 WS-STUDENT-NAME PIC A(15) VALUE 'Tim'.

PROCEDURE DIVISION.
  CALL 'UTIL' USING WS-STUDENT-ID, WS-STUDENT-NAME.
  DISPLAY 'Student Id : ' WS-STUDENT-ID
  DISPLAY 'Student Name : ' WS-STUDENT-NAME
STOP RUN.
```

COBOL Language, Macro, Data Structure, Bug

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1966

Alan Curtis Kay
The Concept of Object-Oriented Programming

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1970

Niklaus Emil Wirth: Pascal Language

Kenneth Lane Thompson: B Language and Unix

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1972

Alan Curtis Kay
Smalltalk

```
m1 := MyClass new.
m2 := MyClass new.
(m1 equals: m2) ifTrue: [
  Transcript show: 'They are equal'
] else: [
  Transcript show: 'They are not equal'
]
```

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1973

Kenneth Lane Thompson
Dennis MacAlistair Ritchie

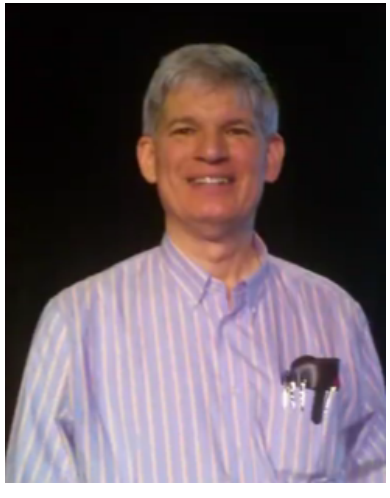
```c
#include <stdio>
int main(void)
{
  printf("Hello, world!\n"
  return 0;
}
```

C Language

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1975

Gerald Jay Sussman
MIT



Guy Lewis Steele Jr.
MIT

```
(define my-counter
  (let ((count 0))
    (lambda ()
      (set! count (+ count 1))
      count)))
```

Scheme Language, Closure

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1983

Bjarne Stroustrup
Nokia Bell Labs

C++ Language

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

- 1992
- The Conference on Functional Programming Languages and Computer Architecture in Portland, Oregon established a special committee to develop the first version of Haskell.
- Lazy Functional Programming Language

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

```haskell
-- Sieve Method for Finding Prime Numbers
primes = filterPrime [2..]
  where
    filterPrime (p : xs) =
      p : filterPrime [x | x ← xs, x `mod` p /= 0]

-- Quick Sort
qsort [] = []
qsort (x : xs) = qsort (filter (≤ x) xs) ++ [x] ++
                 qsort (filter (> x) xs)
```

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

```haskell
-- Sieve Method for Finding Prime Numbers
primes = filterPrime [2..]
  where
    filterPrime (p : xs) =
      p : filterPrime [x | x ← xs, x `mod` p /= 0]

-- Quick Sort
qsort [] = []
qsort (x : xs) = qsort (filter (≤ x) xs) ++ [x] ++
                 qsort (filter (> x) xs)
```

1.4 PROGRAMMING LANGUAGES AND THEIR HISTORY

1995

James Gosling

Java Programming Language

Sun Company

1.5 THE SIMPLEST C++ PROGRAM

```cpp
#include <iostream>
using namespace std;

int main() {
  cout << "Hello, world!" << endl;
  return 0;
}
```

C++ Hello World Program

1.5 THE SIMPLEST C++ PROGRAM

```cpp
#include <iostream>
using namespace std;

int main() {
  cout << "Hello, world!" << endl;
  return 0;
}
```

Header File: from `stdio.h` to `iostream`

1.5 THE SIMPLEST C++ PROGRAM

```cpp
#include <iostream>
using namespace std;

int main() {
  cout << "Hello, world!" << endl;
  return 0;
}
```

Using namespace to avoid name conflicts.

1.5 THE SIMPLEST C++ PROGRAM

```cpp
#include <iostream>
using namespace std;

int main() {
  cout << "Hello, world!" << endl;
  return 0;
}
```

`main` function returns int, 0 means successful.

1.5 THE SIMPLEST C++ PROGRAM

```cpp
#include <iostream>
using namespace std;

int main() {
  cout << "Hello, world!" << endl;
  return 0;
}
```

New I/O Methods.

HOMEWORK

Simple, but Important

Setup C++ Development Environment on Any Platform, Compile and Run HelloWorld Program.