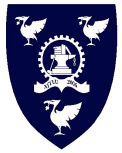


Xi'an Jiaotong-Liverpool University

西交利物浦大學

For Loops and Input-Output

Lab 1

History of C language

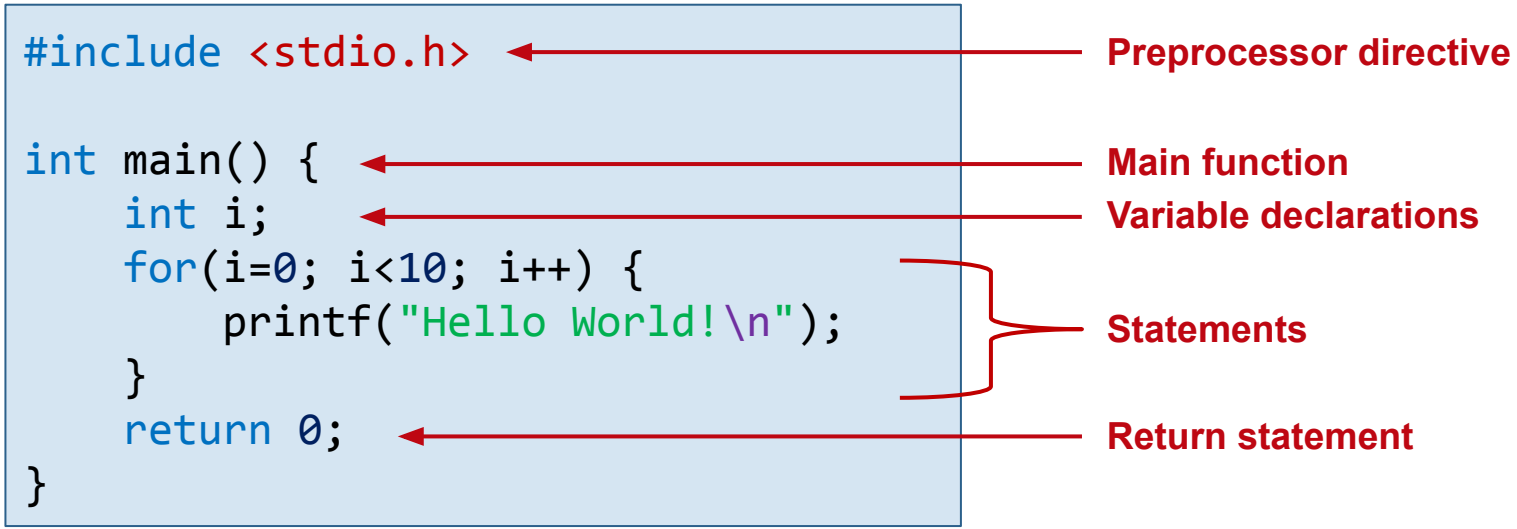
- Dennis Ritchie created the C programming language in 1972 at AT&T (American Telephone & Telegraph) Bell Laboratories in the United States.
- C was originally designed for the UNIX operating system to overcome the shortcomings of preceding languages such as B, BCPL, and others.
- The development of the UNIX operating system began in 1969, and its code was rebuilt in C in 1972.
- Linux kernel development began in 1991, and it is also written in C.
- Low-level language for cross-platform programming, e.g. embedded system.

Language	Year	Developed By
Algol	1960	International Group
BCPL	1967	Martin Richard
B	1970	Ken Thompson
Traditional C	1972	Dennis Ritchie
K & R C	1978	Kernighan & Dennis Ritchie
ANSI C	1989	ANSI Committee
ANSI/ISO C	1990	ISO Committee
C99	1999	Standardization Committee

For more information and latest update, please refer to:
https://en.wikipedia.org/wiki/C_(programming_language)

C Program Structure

- The structure of a simple C program



Repeat instructions

- Print Hello World! in a new line **10 times** with a For Loop

```c
#include <stdio.h>

int main() {
    int i;
    for(i=0; i<10; i++) {
        printf("Hello World!\n");
    }
    return 0;
}
```

declare the counter

start from 0
increment 10x until i = 9
stop

```
for (expr1; expr2; expr3)
    statement
```

Repeat instructions

- Print Hello World! in a new line **10 times** with a For Loop

```c
#include <stdio.h>

int main() {
    int i;
    for(i=0; i<10; i++) {
        printf("Hello World!\n");
    }
    return 0;
}
```

just like our first example,
type this code in **Codecast**,
change it, play with it,
ask question in Forum if
you have some doubts

Comments

- Comment on one line, end of line, multiple lines

```c
#include <stdio.h>

int main() {
    // comment out one printf below
    // printf("One\n");
    printf("Two\n"); // not this one

    /* we can write
    multiple line comments
    this way */

    return 0;
}
```

Print whole numbers

- Use format specifier %d

```c
#include <stdio.h>

int main() {

    printf("Total price of %d apples, each %d yuan,
            is %d yuan\n", 5, 2, 5*2);

    return 0;
}
```

the numbers are inserted into the string, replacing the %d in the same order.

copy paste code in Codecast, change the number, add more numbers...

Store integers in variable

- Variable names: use letters, numbers, _ , start with letter, case sensitive, do not use **reserved words**

such as for, int, return

```
#include <stdio.h>

int main() {
    int numApples, price, totalPrice;
    numApples = 5;
    price = 2;
    totalPrice = numApples * price;
    printf("Total price of %d apples, each %d yuan,
            is %d yuan\n", numApples, price, totalPrice);
    return 0;
}
```

replacing the %d in same order

Read integer user input

- Write a program that ask the price of an apple and then print it
  - Use function `scanf` and operator ampersand `&price`

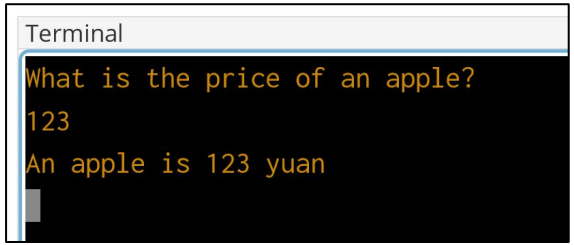
```c
#include <stdio.h>

int main() {

    int price;
    printf("What is the price of an apple?\n");
    scanf("%d", &price);
    printf("An apple is %d yuan\n", price);

    return 0;
}
```

it will read digits from the input, until encounter a whitespace character (enter, spacebar, tab), and store it in variable price

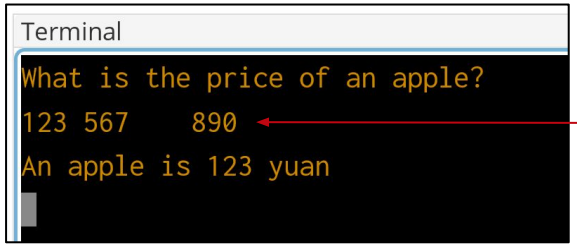
compile and run in Codecast now!

Input by Typing in Interactive Terminal

- If you type 123 and then Enter:

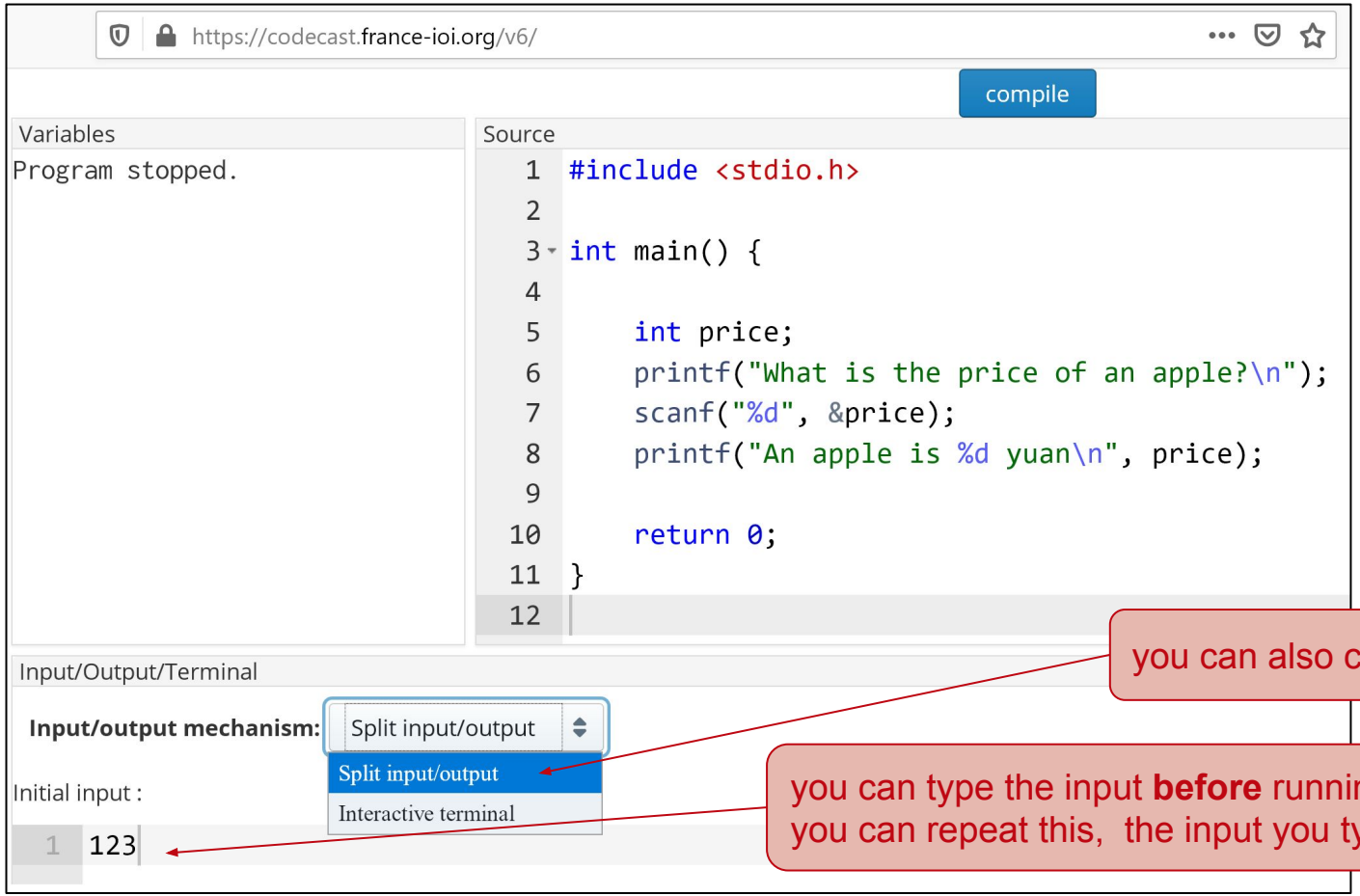


- If you type 123 567 890 and then Enter:



the digits that are read into a number are only the ones before a whitespace character!

Another Way of Input in Codecast



Read multiple integers

- Write a program that reads three numbers and then print it

```c
#include <stdio.h>

int main() {

    int first, second, third;
    scanf("%d %d %d", &first, &second, &third);
    printf("You entered %d, %d, and %d\n",
            first, second, third);

    return 0;
}
```

you can also just write "%d%d%d" the program will still read digits separated by whitespaces

Try running this in Codecast using:

1. Interactive terminal, and
2. Split input/output

Exercise Multiplication Table

- Write a program that reads a number and then print its multiplication table
  - Test case 1 :

    Input: `5`

    Output:

```
0x5 = 0
1x5 = 5
2x5 = 10
3x5 = 15
4x5 = 20
5x5 = 25
6x5 = 30
7x5 = 35
8x5 = 40
9x5 = 45
10x5 = 50
```

WARNING: Hints to the exercise on the next slide

Please try to solve the exercise by yourself first...

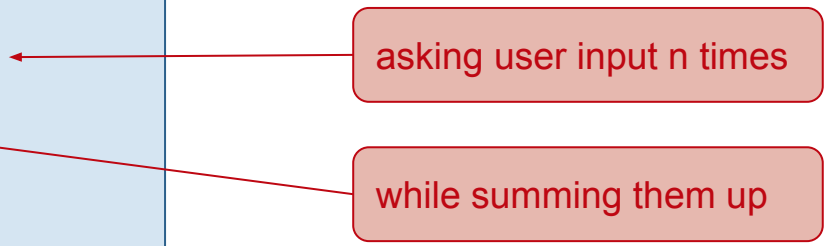
Exercise Multiplication Table Hints

- Write a program that reads a number and then print its multiplication table
  - Declare a variable num and a counter i
  - Use scanf and %d to read user input, store it to &num
  - Use for and counter i to loop 11 times
    - and repeatedly printf i, num, and i times num using three %d

Reading inside a loop

- Write a program that reads a number n, ask the user to enter n numbers, and then print the sum of those n numbers

```c
#include <stdio.h>
int main() {
    int i, n, num;
    int sum = 0;
    scanf("%d", &n);
    for(i=0; i<n; i++) {
        scanf("%d", &num);
        sum += num;
    }
    printf("%d\n", sum);
    return 0;
}
```



Declare, assign, and print characters

- Use format specifier %c

```c
#include <stdio.h>

int main() {

    char letter;
    letter = 'a';
    printf("The letter is %c\n", letter);

    return 0;
}
```

Read characters

- Do **not** separate by space

```c
#include <stdio.h>

int main() {

    char letter1, letter2;
    printf("Enter two letters: ");
    scanf("%c%c", &letter1, &letter2);
    printf("The letters are %c and %c.\n", letter1, letter2);

    return 0;
}
```

if you use space, instead of storing the second letter, you will store the space. try it !!

Exercise Char Tree

- Write a program that reads an input character and displays the tree pattern
- Test case 1 :

  Input:
  #

  Output:
  ++++#+++++
  +++#####+++
  ++#######++
  +########+
  #########

WARNING: Hints to the exercise on the next slide

Please try to solve the exercise by yourself first...

Exercise : Char Tree Hints

- Write a program that reads an input character and displays the tree pattern
  - Declare a char variable
  - Read an input character by scanf and %c
  - Use 5 printf and multiple %c to print the tree
    - you can use for, but you don't have to

Declare, assign, and print decimal numbers

- Use format specifier `%lf`
  - Default 6 digits after point, if want 2 digits, use `%.2lf`

```c
#include <stdio.h>

int main() {

    double height;
    height = 1.72;
    printf("Your height is %.2lf\n", height);

    return 0;
}
```

Read decimal numbers

- Use format specifier `%lf`

just like reading and printing whole number or character, but use %lf and . followed by a number to format the precision you want

```c
#include <stdio.h>

int main() {

    double height;
    scanf("%lf", &height);
    printf("Your height is %.2lf\n", height);

    return 0;
}
```

Read, assign, and print integers and doubles

- Use format specifier %d and %lf

```
#include <stdio.h>

int main() {

    int age;
    double height;
    scanf("%d%lf", &age, &height);
    printf("Your age is %d and your height is %.2lf\n",
            age, height);

    return 0;
}
```

Number division in C

- Integer division and floating point division

```c
#include <stdio.h>

int main() {

    printf("5/2 equals %d\n", 5/2);
    printf("5.0/2.0 equals %lf\n", 5.0/2.0);
    printf("5/2.0 equals %lf\n", 5/2.0);
    printf("5.0/2 equals %lf\n", 5.0/2);

    return 0;
}
```

try in Codecast.
which one you use if you want

1. truncated result?
2. decimal result?

Remainder in integer division

- Use operator modulo %

starting to get tricky!
must distinguish when % is used as format specifier, or as a modulo operator!

```
#include <stdio.h>

int main() {

    int number, divisor;
    scanf("%d%d", &number, &divisor);
    printf("%d divided by %d is %d\n", number, divisor,
            number/divisor);
    printf("The remainder is %d\n", number%divisor);

    return 0;
}
```

Convert integers to/from doubles

- Use casting `(double)` and `(int)`

```c
#include <stdio.h>

int main() {

    int num1, num2;
    double dec1, dec2;
    scanf("%d %lf", &num1, &dec2);
    dec1 = (double) num1;
    num2 = (int) dec2;
    printf("dec1 = %lf\n", dec1);
    printf("num2 = %d\n", num2);

    return 0;
}
```

try and experiment in Codecast! what happen with the fractional part after the decimal point of your input?

Exercise Average Grades

- Write a program that computes the average grades
- Your program first read an integer indicating the number of grades to be averaged. Next, read the grades one by one, all of which are integers as well. Finally, calculate and print the average of the grades **to two decimal places**.
- Test case 1 :

  Input:

  ```
  3
  5
  10
  2
  ```

  Output:

  ```
  5.67
  ```

WARNING: Hints to the exercise on the next slide

Please try to solve the exercise by yourself first...

Exercise Average Grades Hints

- Write a program that computes the average grades
  - Declare integer variables for the counter, the grade, how many grades to read
  - Declare and initialize an integer variable for the sum to 0
  - Read the number of grades with scanf
  - Loop that many times
    - Read the grade with scanf
    - Add it to sum
  - Cast sum to double and compute the average with (decimal) division, inside a printf that prints it using ".2lf\n"