Computer Systems & Low-Level Programming

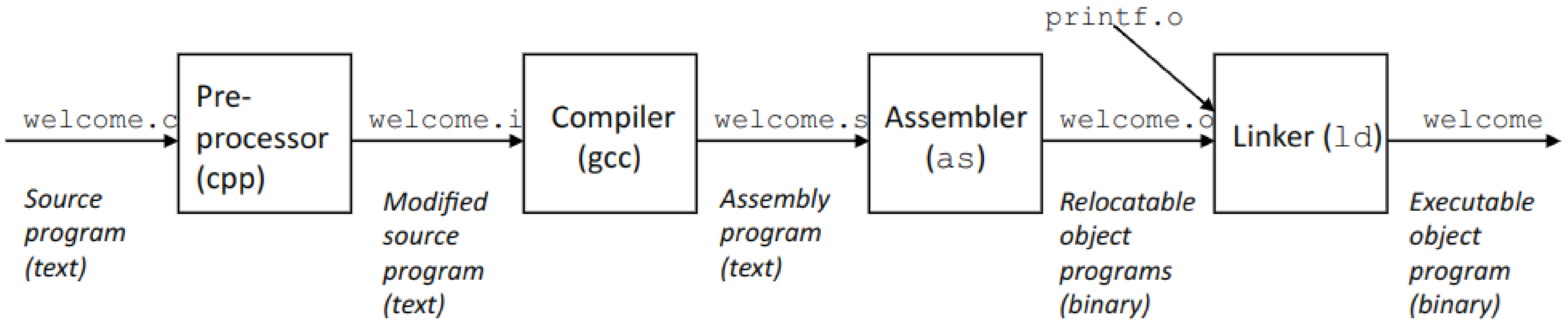
Basics of Systems and C

Marija Stanojevic
Spring 2019

Review

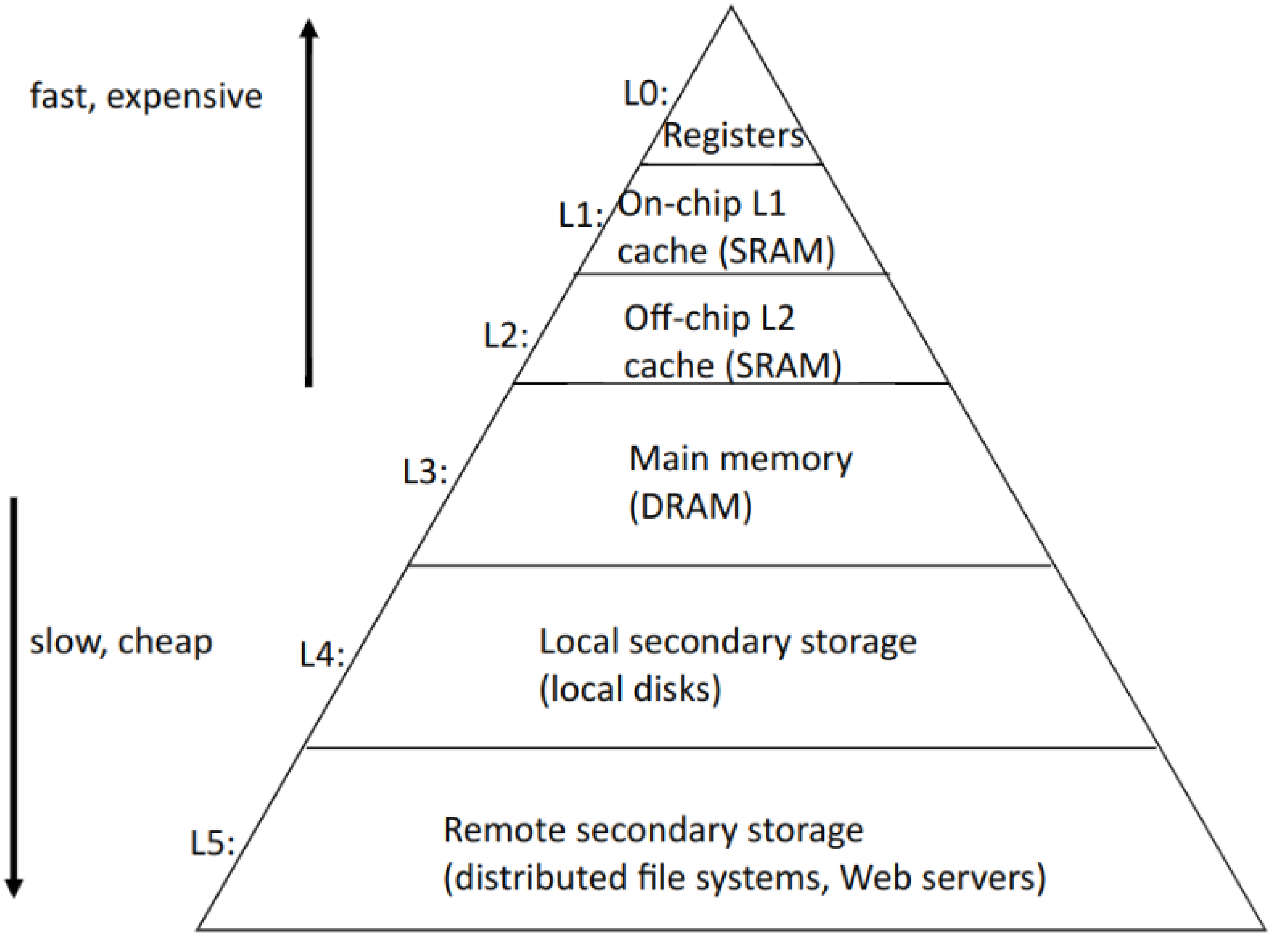
- What does typed language mean? Is C typed language?
    - You need to define type of variable (int, float, double, char). C is typed language
- Why is C important/used today?
    - It’s fast and tiny. It is part of OS, microchip prog., IoT, embedded and real-time systems.
- What is compiler doing? What is linker doing?
    - Compiler translates from C into assembler. Linker links assembler code from different files.
- What is syntax error (compile-time error)? Examples?
    - Compiler can’t recognize statement because it violates language rules (e.g. forgotten ;)
- What is run-time error? Examples?
    - Runned program is in forbidden position or doing undefined operation (e.g. division with 0)
- What is assembly language? How does it differ from C?
    - Lower level than C. Each word of assembler translates into 0s and 1s.
- What is interpreter? Which languages are interpreted?
    - Interpreter executes high-level languages. Much slower than compiler. More on next slide.

Interpreter	Compiler
Translates program one statement at a time.	Scans the entire program and translates it as a whole into machine code.
It takes less amount of time to analyze the source code but the overall execution time is slower.	It takes large amount of time to analyze the source code but the overall execution time is comparatively faster.
No intermediate object code is generated, hence are memory efficient.	Generates intermediate object code which further requires linking, hence requires more memory.
Continues translating the program until the first error is met, in which case it stops. Hence debugging is easy.	It generates the error message only after scanning the whole program. Hence debugging is comparatively hard.
Programming language like Ruby PHP JAVA Perl R Powershell	Programming language like C C++ C# Objective-C SWIFT Fortran.

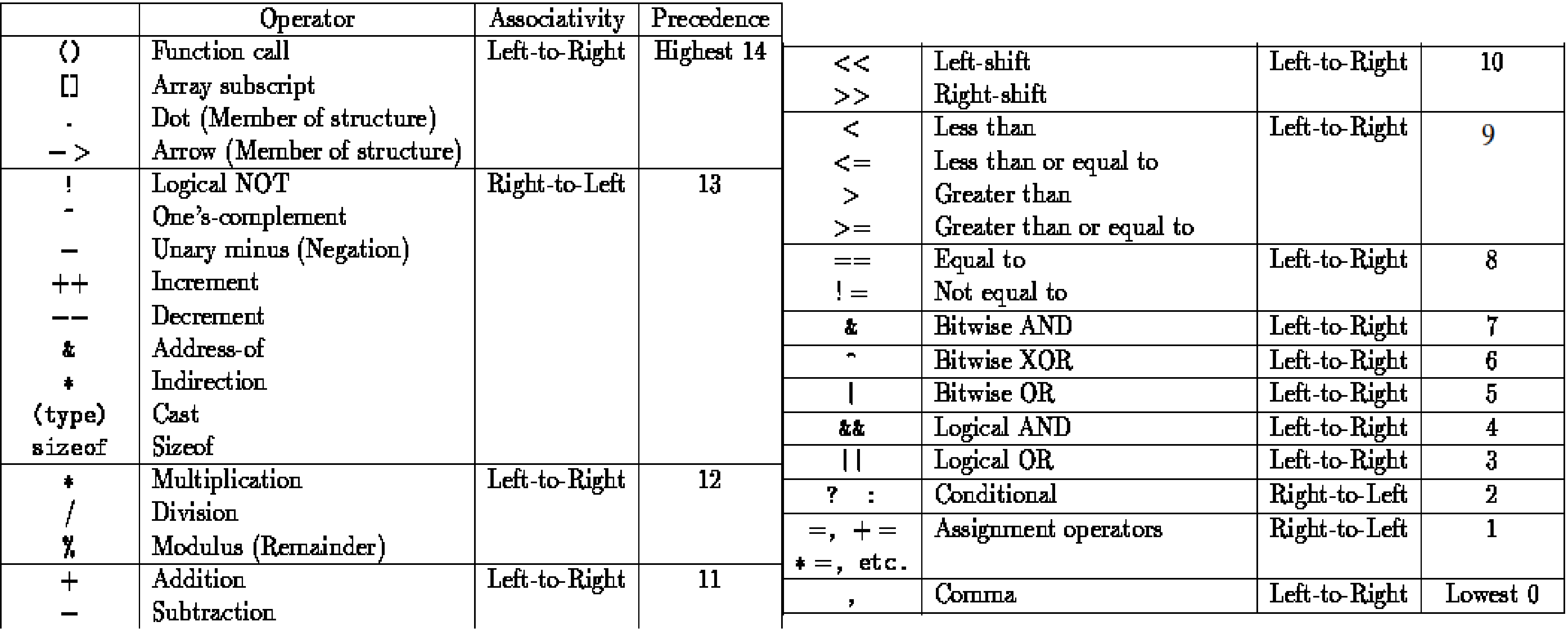


- **Preprocessing:** `gcc –E –o welcome.i welcome.c`
- **Compiling:** `gcc –S –o welcome.s welcome.i`
- **Assembling:** `gcc –c –o welcome.o welcome.s`
- **Linking:** `gcc –o welcome welcome.o`

memory hierarchy


fast, expensive
slow, cheap
L0: Registers
L1: On-chip L1 cache (SRAM)
L2: Off-chip L2 cache (SRAM)
L3: Main memory (DRAM)
L4: Local secondary storage (local disks)
L5: Remote secondary storage (distributed file systems, Web servers)

Operators in C



	Operator	Associativity	Precedence
() [] . ->	Function call Array subscript Dot (Member of structure) Arrow (Member of structure)	Left-to-Right	Highest 14
! ~ - ++ -- & * (type) sizeof	Logical NOT One's-complement Unary minus (Negation) Increment Decrement Address-of Indirection Cast Sizeof	Right-to-Left	13
* / %	Multiplication Division Modulus (Remainder)	Left-to-Right	12
+ -	Addition Subtraction	Left-to-Right	11
<< >>	Left-shift Right-shift	Left-to-Right	10
< <= > >=	Less than Less than or equal to Greater than Greater than or equal to	Left-to-Right	9
== !=	Equal to Not equal to	Left-to-Right	8
&	Bitwise AND	Left-to-Right	7
^	Bitwise XOR	Left-to-Right	6
|	Bitwise OR	Left-to-Right	5
&&	Logical AND	Left-to-Right	4
||	Logical OR	Left-to-Right	3
? :	Conditional	Right-to-Left	2
=, += *=, etc.	Assignment operators	Right-to-Left	1
,	Comma	Left-to-Right	Lowest 0

Keywords

auto	do	goto	signed	unsigned
break	double	if	sizeof	void
case	else	int	static	volatile
char	enum	long	struct	while
const	extern	register	switch	
continue	float	return	typedef	
default	for	short	union	

Keywords added in C99 standard

_Bool   _Complex   _Imaginary   inline   restrict

Keywords added in C11 standard

_Alignas   _Alignof   _Atomic   _Generic   _Noreturn   _Static_assert   _Thread_local

Data types

- char, 1B, numbers between -128 and 127 (signed char) - ASCII table
- unsigned - just positive numbers (add in front of any whole number data type)
- unsigned char, 1B, numbers between 0 and 255
- short, 2B, numbers between -32,768 and +32,767
- int, 4B, -2,147,483,648 to +2,147,483,647
- long, 4B, -2,147,483,648 to +2,147,483,647
- float, decimal numbers with 4B
- double, decimal numbers with 8B
- long double, decimal numbers with 10B
- void - for functions that don’t return anything, address when type is not known
- int *, short*, double*, char* to represent address of certain type

Code from Lab 2

```
// hello.c, part 1
#include<stdio.h>
int canIvote(int age) {
	if (age >= 21) {
		return 1;
	} else {
		return 0;
	}
}
int main() {
	char name[50];
	printf("Address %p", name);
	int age;
	float grade;
	char exclamation = '!';
	scanf("%s", name);
	printf("Hello %s%c What's your
age?\n", name, exclamation);
```

```
// part 2 of file
	scanf("%d", &age);
	printf("What is your avg. grade?");
	scanf("%f", &grade);
	if (age > 0) {
		printf("Age is correct! \n");
	} else {
		printf("Error! You didn't write correct age\n");
		age = 0;
	}
	printf("Your age is %d. Your avg. grade is %.2f \n",
age, grade);
	for (int i=0; i < 16; i++) {
		printf("Still too young to drive \n");
	}
	int vote = canIvote(age);
	printf("Voting %d", vote);
	return 0;
}
```