EASTERN WASHINGTON UNIVERSITY
start something big

Compare C with Java

Computer Science Department
Eastern Washington University
Yun Tian (Tony) Ph.D.

Review

- Number Systems
  - Unsigned notation
  - Signed magnitude notation
  - Two's Complement notation
- Data representations
  - characters strings
  - text files and binary files
  - Image representation

Outline for Today

- Compare C with Java

C vs. Java

Items	C	Java
Type of language	Function Oriented	Object Oriented
Basic programming unit	Function	Class = ADT
Portability of source code	Yes, cross platform	Yes
Portability of compiled code	No, recompile for each architecture	Yes, JVM

C vs. Java

Items	C	Java
Safety	Limited, e.g. we can access array element beyond boundary, but **error prone**!	Built-in
Compilation	gcc hello.c –o hello, to create machine native code	javac Hello.java creates Java virtual machine language bytecode

C vs. Java

Items	C	Java
Using Math library	gcc -lm calculate.c Explicitly load math library using flag –lm when compile and link.	Built-in, no specific flags
joint compilation	gcc main.c helper1.c helper2.c -o myapp	javac Main.java any dependent files are automatically re-compiled if needed.

C vs. Java

Items	C	Java
execution	./myapp (think how to remove "./" ?)	java Hello interprets byte code

hello, world	#include<stdio.h>
	int main(void)
	{
	printf("Hello\n");
	return 0;
	}
		public class HelloWorld
		{
		public static void main(String[] args)
		{
		System.out.println("Hello");
		}
		}

C vs. Java

Items	C	Java
integer	Usually 32 bit 2's complement; int a = 10;	32 bit 2's complement; int a = 10;
long int	Usually 32-bit, but may 64-bit on some platform. long a = 1009L;	64-bit long along=1L;
long long int	64-bit long long allong = 1LL;	Not applicable

C vs. Java

Items	C	Java
float	Usually 32 bit Usually IEEE 754	IEEE 754 binary floating point.
double	64-bit Usually IEEE754	64-bit IEEE 754 representation

On different platforms, such as embedded devices, the size of float and double may vary. Please refer to the float.h file on that platform.

C vs. Java

Items	C	Java
boolean	Use integer 0 (zero) for false, Nonzero for true.	boolean is own type. boolean b = true; true, false.
character type	char is 8 bit ASCII char c = 'k';	char is 16 bit UNICODE

C vs. Java

Items	C	Java
for loops	int i; for (i = 0; i < 100; i++)	for (int i = 0; i < 100; i++)
Declare static array	int d[10]; Create array d that can hold 10 integers at most. The number inside [ ] should be a CONSTANT.	Not applicable
Create dynamic array	int *a = malloc(numElements * sizeof(int));	int[] a = new int[numElements];

C vs. Java

Items	C	Java
string	'\0' terminated 1D character array	Array built-in immutable String data type
accessing a library	#include <stdio.h>	import java.io.File;

C vs. Java

Items	C	Java
string	'\0' terminated 1D character array	built-in immutable String data type
accessing a library	#include <stdio.h>	import java.io.File;

C vs. Java

Items	C	Java
memory address	pointer	reference
manipulating pointers	***, &, +**	**No direct manipulation permitted**
functions	int max(int a, int b)	public static int max(int a, int b)

C vs. Java

Items	C	Java
allocating memory	malloc	new
De-allocating memory	Programmer has to manually free what he/she has allocated.	Automatic garbage collection takes care of no-longer useful memory.
variable auto-initialization	not guaranteed. No compile error if you forget initializing.	instance variables (and array elements) initialized to 0, null, or false, compile-time error to access uninitialized variables

C vs. Java

Items	C	Java
Data type for generic item	void *	Object
Demotions Convert long to int	Automatic, but give warning and may lose precision.	Must explicitly cast,
null	NULL	null

C vs. Java

Items	C	Java
Overloading	No same function name is allowed. Good practice to use function names different from variable names.	Yes for methods

Variable declaration	At beginning of a block
	int airSpeed = 0;
	int air_speed = 0;
		Before you use it
		int airSpeed = 0;

commenting	/* code block */ // inline comment	Same as C

Take Home Summary

- Safety is limited in C,
  - So be **very** cautious when deal with memory access.
    - including allocating, free, array boundary, initialization, pointer to unallocated space etc.
- No overloading in C,
  - function name should be unique.
  - Good practice ***not*** to name function same as variable name.

Take Home Summary

- When compiling C code, enable all warnings.

  gcc –Wall source1.c source2.c –o myProg

- **Getting rid of all warnings in you code saves millions of hours of debugging logic error later.**

Tomorrow

- C types
- Simple C programs