A Fast Review of C Essentials
Part I

Structural Programming
by
Z. Cihan TAYSI

Outline

- Program development
- C Essentials
  - Functions
  - Variables & constants
  - Names
  - Formatting
  - Comments
  - Preprocessor
- Data types
- Mixing types

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Program Development

source file
source file
source file

object file
object file
object file

Runtime Library

link

Executable Code

- The task of compiler is to translate source code into machine code
- The compiler's input is **_source code_** and its output is **_object code_**.
- The linker combines separate object files into a single file
- The linker also links in the functions from the runtime library, if necessary.
- Linking usually handled automatically.
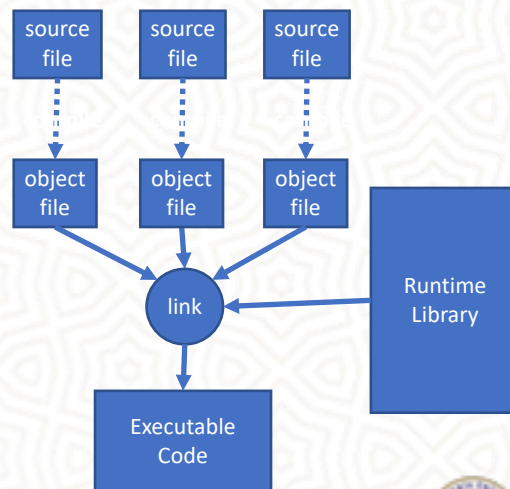
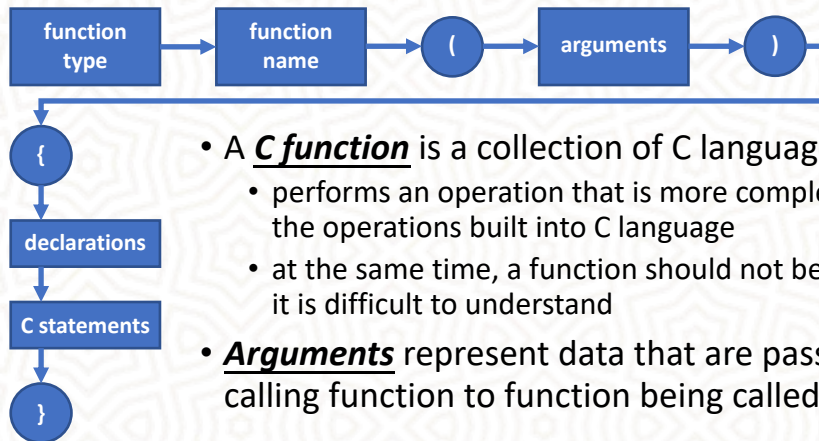*Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü*

Program Development CONT'D

- One of the reasons C is such a small language is that it defers many operations to **_a large runtime library._**
- The runtime library is a collection of object files
  - Each file contains the machine instructions for a function that performs one of a wide variety of services
    - The functions are divided into groups, such as I/O, memory management, mathematical operations, and string manipulation.
  - For each group there is a source file, called a **_header file_**, that contains information you need to use these functions
    - by convention , the names for header files end with *.h* extention
- For example, one of the I/O runtime routines, called **_printf()_**, enables you to display data on your terminal. To use this function you must enter the following line in your source file
  - #include <stdio.h>

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

C Essentials

```
[function type] → [function name] → ( → [arguments] → )
      ↓
     { 
      ↓
[declarations]
      ↓
[C statements]
      ↓
     }
```

- A **C function** is a collection of C language operations.
  - performs an operation that is more complex than any of the operations built into C language
  - at the same time, a function should not be so complex that it is difficult to understand
- **Arguments** represent data that are passed from calling function to function being called.

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Variables & Constants

- The statement
  - j = 5 + 10;
- **A constant** is a value that never changes
- **A variable** achieves its variableness by representing a location, **or address**, in computer memory.

Variable	Address	Contents
		4 bytes
	2482	
j	2486	15
	2490	

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Names

- In the C language, you can name just about anything
  - variables, constants, functions, and even location in a program.
- Names may contain
  - letters, numbers, and the underscore character (_)
  - **but must start with a letter or underscore**…
- The C language is ___case sensitive___ which means that it differentiates between lowercase and uppercase letters
  - VaR, var, VAR
- A name **can NOT be** the same as one of the ___reserved keywords___.

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Names cont'd

- **LEGAL NAMES**
  - j
  - j5
  - __system_name
  - sesquipedalial_name
  - UpPeR_aNd_LoWeR_cAsE_nAmE

- **ILLEGAL NAMES**
  - 5j
  - $name
  - int
  - bad%#*@name

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Names cont'd

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

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Expressions

- An ***expression*** is any combination of operators, numbers, and names that donates the computation of a value.
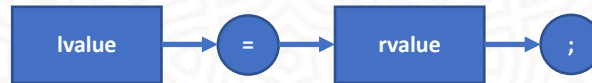- **Examples**
  - 5   A constant
  - j   A variable
  - 5 + j   A constant plus a variable
  - f()   A function call
  - f()/4   A function call, whose result is divided by
  -       a constant

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Assignment Statements

| lvalue | = | rvalue | ; |

- The left hand side of an assignment statement, called an **lvalue**, must evaluate to memory address that can hold a value.
- The expression on the right-hand side of the assignment operator is sometimes called an **rvalue**.
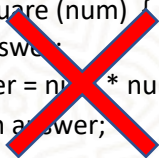
answer = num * num; ✓

num * num = answer; ✗

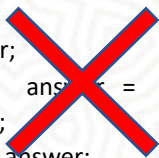*Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü*

Formatting Source Code

```
int square (num) {
int answer;
answer = num * num;
return answer;
}

int square (num)
{ int
answer;
        answer =        num
* num;
return answer;
}
```
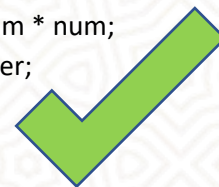✗

```
int square (num)  {
    int answer;
    answer = num * num;
    return answer;
}
```
✓

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Comments

- A comment is text that you include in a source file to explain what the code is doing!
  - Comments are for human readers – compiler ignores them!
- The C language allows you to enter comments between the symbols /* and */
- Nested comments are NOT supported
- **What to comment ?**
  - Function header
  - changes in the code

```
/* square()
 * Aurhor : P. Margolis
 * Initial coding : 3/87
 * Params : an integer
 * Returns : square of its parameter
 */
```

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

The main() Function

```
int main ( ) {
        extern int square();
        int solution;
        solution = square(5);
        exit(0);
}
```

- The **_exit()_** function is a runtime library routine that causes a program to end, returning control to operating system.
  - If the argument to exit() is zero, it means that the program is ending normally without errors.
  - Non-zero arguments indicate abnormal termination of the program.
- Calling exit() from a main() function is exactly the same as executing **_return_** statement.

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

printf() and scanf() Functions

- The printf() function can take any number of arguments.
  - The first argument called the ***format string***. It is enclosed in double quotes and **may contain** text and ***format specifiers***
- The scanf() function is the mirror image of printf(). Instead of printing data on the terminal, it reads data entered from keyboard.
  - The first argument is a format string.
  - **The major difference between scanf() and printf() is that the data item arguments must be lvalues**

```
int num;

scanf("%d", &num);
printf("num : %d\n", num);
```

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Preprocessor

- The preprocessor executes automatically, when you compile your program
- All preprocessor directives begin with pound sign (#), which must be the first non-space character on the line.
  - unlike C statements a preprocessor directive ends with a newline, **NOT a semicolon**
- It is also capable of
  - macro processing
  - conditional compilation
  - debugging with built-in macros

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Preprocessor cont'd

- The **_define_** facility
  - it is possible to associate a name with a constant
    - #define NOTHING 0
  - It is a common practice to all uppercase letters for constants
  - naming constants has two important benefits
    - it enable you to give a descriptive name to a nondescript number
    - it makes a program easier to change
  - be careful NOT to use them as variables
    - **NOTHING = j + 5**

Preprocessor cont'd

- The **_include_** facility
  - #include directive causes the compiler to read source text from another file as well as the file it is currently compiling
  - the #include command has two forms
    - #include <filename>
      - **the preprocessor looks in a special place designated by the operating system. This is where all system include files are kept.**
    - #include "filename"
      - **the preprocessor looks in the directory containing the source file. If it can not find the file, it searches for the file as if it had been enclosed in angle brackets!!!**

hello world!!!

#include <stdio.h>   • include standard input output library

int main (void) {   • start point of your program

    printf("Hello World...\n");
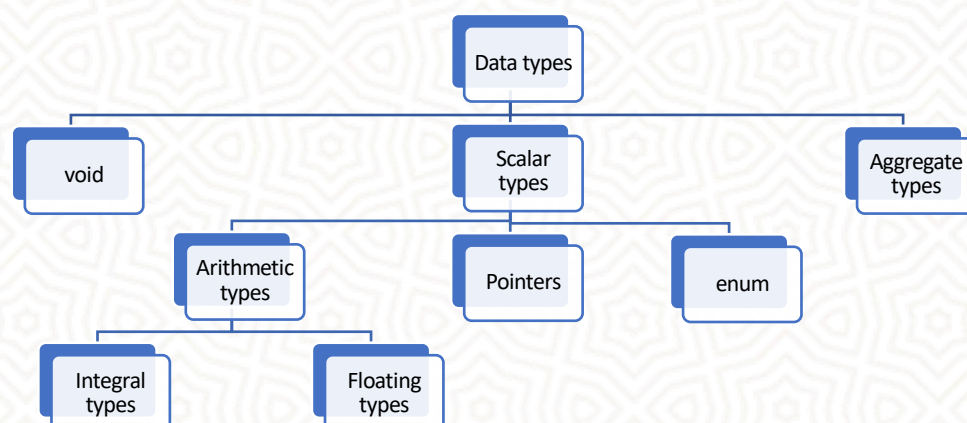   return 0;   • return a value to calling program
}   • in this case 0 to show success?

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Data Types

```
                        Data types
                            |
        +-------------------+-------------------+
        |                   |                   |
      void            Scalar types        Aggregate types
                            |
          +-----------------+-----------------+
          |                 |                 |
    Arithmetic           Pointers           enum
       types
     +-----+-----+
     |           |
  Integral    Floating
   types       types
```

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Data Types cont'd

- There are 9 reserved words for scalar data types
- Basic types
  - char, int, float, double, enum
- Qualifiers
  - long, short, signed, unsigned
- To declare j as an integer
  - int j;
- You can declare variablers that have the same type in a single declaration
  - int j,k;
- ***All declaretions in a block must appear before any executable statements***

char	double	short	signed
int	enum	long	unsigned
float			

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Different Types of Integers

- The only requirement that the ANSI Standard makes is that a byte must be ***at least 8 bits long***, and that ints must be ***at least 16 bits long*** and must represent the "***natural***" size for computer.
  - natural means the number of bits that the CPU usually handles in a single instruction

Type	Size (in bytes)	Value Range
int	4	-2^{31} to $2^{31}-1$
short int	2	-2^{15} to $2^{15}-1$
long int	4	-2^{31} to $2^{31}-1$
unsigned short int	2	0 to $2^{16}-1$
unsigned long int	4	0 to $2^{32}-1$
signed char	1	-2^{7} to $2^{7}-1$
unsigned char	1	0 to $2^{8}-1$

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Different Types of Integers cont'd

- Integer constants
  - Decimal
  - Octal
  - Hexadecimal
- In general, an integer constant has type int, if its value can fit in an int. Otherwise it has type long int.
- Suffixes
  - u or U
  - l or L

Decimal	Octal	Hexadecimal
3	003	0x3
8	010	0x8
15	017	0xF
16	020	0x10
21	025	0x15
-87	-0127	-0x57
255	0377	0xFF

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Floating Point Types

- to declare a variable capable of holding floating-point values
  - **float**
  - **double**
- The word ***double*** stands for double-precision
  - it is capable of representing about twice as much precision as a ***float***
  - A float generally requires ***4 bytes***, and a double generally requires ***8 bytes***
  - ***read more about limits in <limits.h>***

- Decimal point
  - 0.356
  - 5.0
  - 0.000001
  - .7
  - 7.
- Scientific notation
  - 3e2
  - 5E-5

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Initialization

- A declaration allocates memory for a variable, but it does not necessarily store an inital value at the location
  - ***If you read the value of such a variable before making an explicit assignment, the results are unpredictable***
- To initialize a variable, just include an assignment expression after the variable name
  - char ch = 'A' ;
- It is same as
  - char ch;
  - ch = 'A';

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü
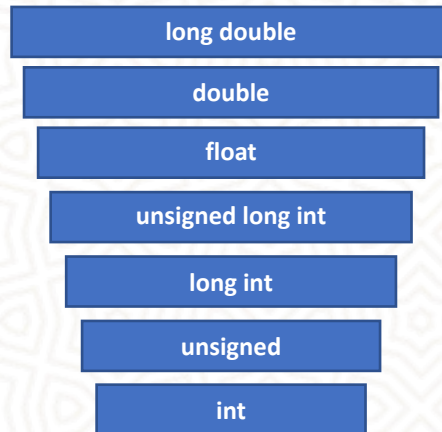
Mixing Types

- Implicit convertion
- Mixing signed and unsigned types
- Mixing integers with floating point types
- Explicit conversion

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Mixing Types cont'd

long double
double
float
unsigned long int
long int
unsigned
int

Implicit Conversions

- When the compiler encounters an ***expression***, it divides it into ***subexpressions***, where each expression consists of one operator and one or more objects, called ***operands***, that are **bound to the operator**.
- **Ex :** − 3 / 4 + 2.5   # The expression contains three operators −, /, +
- Each operator has its own rules for operand type agreement, but most binary operators require both operands to have the same type.
  - If the types differ, the compiler converts one of the operands to agree with the other one.
  - For this conversion, compiler resorts to the hierarchy of data types. **(Please remember previous slide)**
- **Ex :** 1 + 2.5   # involves two types, an int and a double

Mixing Signed and Unsigned Variables

- The only difference between signed and unsigned integer types is the way they are interpreted.
  - They occupy same amount of storage
- 11101010
  - has a decimal value of -22 (in two's complement notation)
  - An unsigned char with the same binary representation has a decimal value of 234
- 10u – 15 = ?
  - – 5
  - 4,294,967,291

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Mixing Integers with Floating Types

- Invisible conversions int j;

                                float f;

                                j + f ;   // j is converted to float

                                j + f + 2.5;   // j and f both converted to double

- **Loss of precision**   j = 2.5;   // j's value is 2

                                j = 999999999999.888888   // overflow

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Explicit Conversions - Cast

int j=2, k=3;
float f;

f = k / j ;

f = (float) k / j;

- **Explicit conversion is called casting and is performed with a construct called _a cast_**

- **To cast an expression, enter the target data type enclosed in parenthesis directly before expression**

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

Enumeration Data Type

enum { red, blue, green, yellow } color;
enum { bright, medium, dark } intensity;

color = yellow;   // OK
color = bright;   // Type conflict
intensity = bright;   // OK
intensity = blue;   // Type conflict
color = 1;   // Type conflict
color = green + blue;   // Misleading usage

- **_Enumeration types_** enable you to declare variables and the set of named constants that can be legally stored in the variable.

- The default values start at zero and go up by one with each new name.

- You can override default values by specifying other values

Yıldız Teknik Üniversitesi - Bilgisayar Mühendisliği Bölümü

void Data Type

- The void data type has two important purposes.

- The first is to indicate that a function does not return a value
  - void func (int a, int b);

- The second is to declare a generic pointer
  - **We will discuss it later !**

typedef

- *typedef* keyword lets you create your own names for data types.
- Semantically, the variable name becomes a synonym for the data type.
- By convention, typedef names are capitalized.

typedef long int INT32;

long int j;
INT32 j;