C Fundamentals

HK Rana

Writing a Simple Program

- PRINT A MESSAGE, “TO C, OR NOT TO C: THAT IS THE QUESTION.”

Comments

- Why Important?
- Comment using “//”
- Comment using “/* */”

Variables

- Why we need variables?

- Types of data or data types

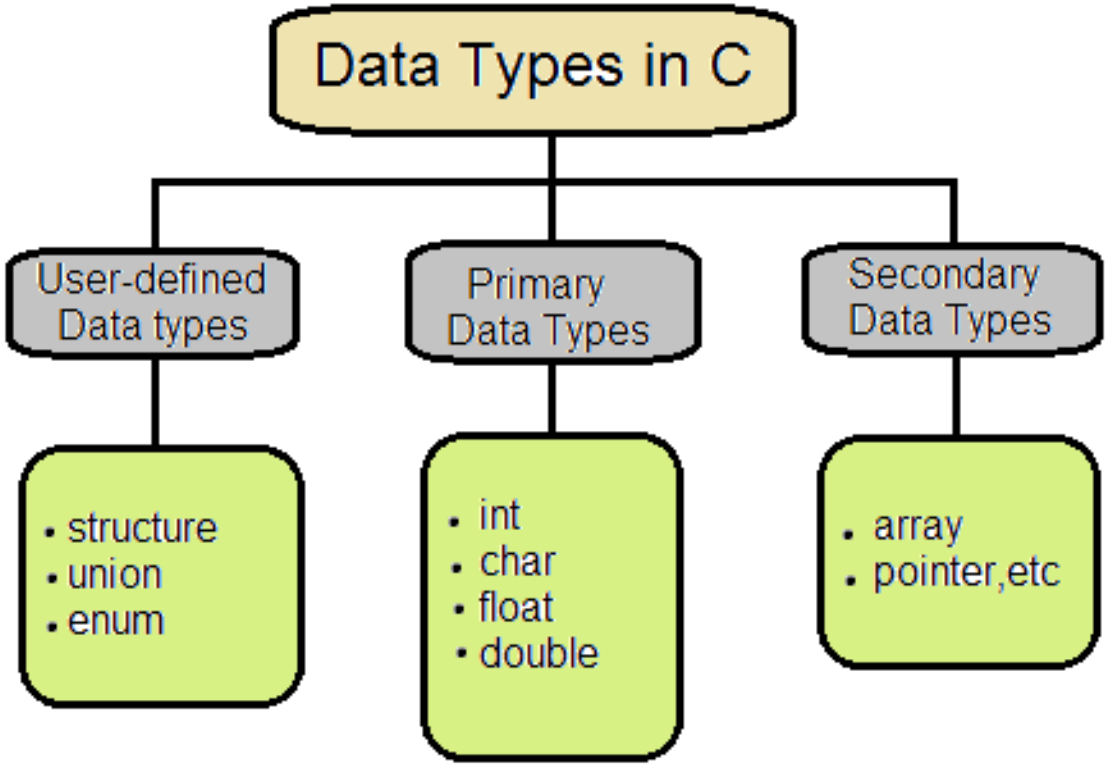
- Why we have to declare variables?

- Value assignment to a variable

- Value initializing to a variable

** Printing variable values **

Data Types in C



Rules for naming variables:

1. A valid variable can have letters (both uppercase and lowercase letters), digits and underscores.
2. The first letter of a variable should be either a letter or an underscore.
3. You cannot use keywords as variable.
4. Blank spaces are not allowed within a variable.
5. Variable should not be of length more than 31 characters.

Size is matter for a variable.

Specifier , Size and Range of Data Types:

Type	Format Specifier	Storage size (bits)	Value range
char	%c	8	-128 to 127
unsigned char	%uc	8	0 to 255
int	%d or %i	16	-32,768 to 32,767
unsigned int	%ud	16	0 to 65,535
long int	%ld	32	-2,147,483,648 to 2,147,483,647
unsigned long	%uld	32	0 to 4,294,967,295
float	%f	32	3.4E-38 to 3.4E+38
double	%lf	64	1.7E-308 to 1.7E+308

A Sample Program

Computing the Dimensional Weight of a Box

Shipping companies don't especially like boxes that are large but very light, since they take up valuable space in a truck or airplane. In fact, companies often charge extra for such a box, basing the fee on its volume instead of its weight. In the United States, the usual method is to divide the volume by 166 (the allowable number of cubic inches per pound). If this number—the box's "dimensional" or "volumetric" weight—exceeds its actual weight, the shipping fee is based on the dimensional weight. (The 166 divisor is for international shipments; the dimensional weight of a domestic shipment is typically calculated using 194 instead.)

Let's say that you've been hired by a shipping company to write a program that computes the dimensional weight of a box. Since you're new to C, you decide to start off by writing a program that calculates the dimensional weight of a particular box that's 12″ × 10″ × 8″. Division is represented by / in C, so the obvious way to compute the dimensional weight would be

```
Dimensions: 12x10x8
Volume (cubic inches): 960
Dimensional weight (pounds): 6
```

Reading input from user

- Scanf function

- Defining names of constants

Computing the Dimensional Weight of a Box (Revisited)

Shipping companies don't especially like boxes that are large but very light, since they take up valuable space in a truck or airplane. In fact, companies often charge extra for such a box, basing the fee on its volume instead of its weight. In the United States, the usual method is to divide the volume by 166 (the allowable number of cubic inches per pound). If this number—the box's "dimensional" or "volumetric" weight—exceeds its actual weight, the shipping fee is based on the dimensional weight. (The 166 divisor is for international shipments; the dimensional weight of a domestic shipment is typically calculated using 194 instead.)

Let's say that you've been hired by a shipping company to write a program that computes the dimensional weight of a box. Since you're new to C, you decide to start off by writing a program that calculates the dimensional weight of a particular box that's 12″ × 10″ × 8″. Division is represented by / in C, so the obvious way to compute the dimensional weight would be

PROGRAM **Converting from Fahrenheit to Celsius**

The following program prompts the user to enter a Fahrenheit temperature; it then prints the equivalent Celsius temperature. The output of the program will have the following appearance (as usual, input entered by the user is underlined):

```
Enter Fahrenheit temperature: 212
Celsius equivalent: 100.0
```

The program will allow temperatures that aren't integers; that's why the Celsius temperature is displayed as `100.0` instead of `100`. Let's look first at the entire program, then see how it's put together.

```
celsius = (fahrenheit - FREEZING_PT) * SCALE_FACTOR;
```

```
#define FREEZING_PT 32.0f
#define SCALE_FACTOR (5.0f / 9.0f)
```

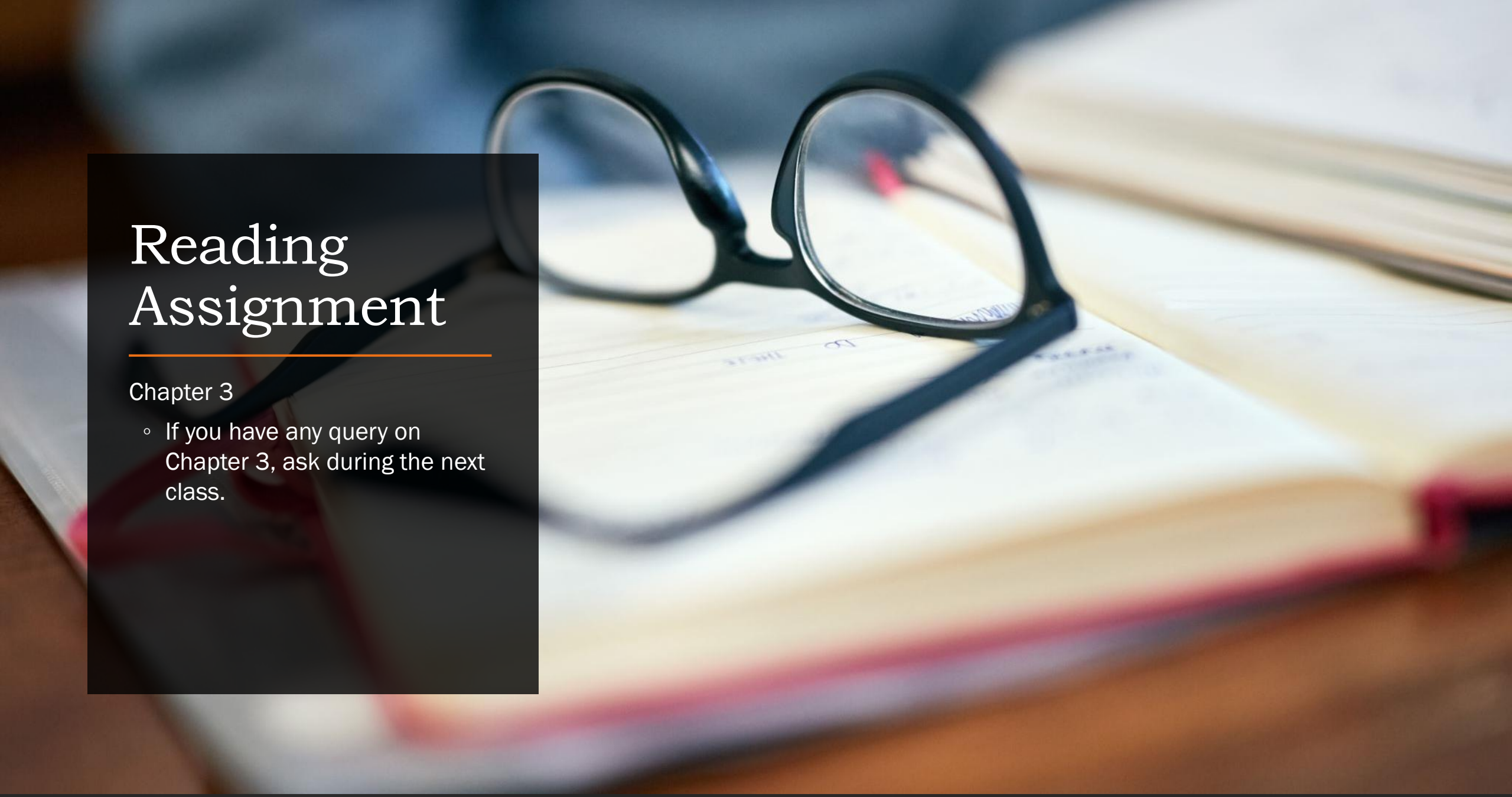
Another Example

Others

- Identifiers
- Keywords

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

†C99 only



Reading Assignment

Chapter 3

- If you have any query on Chapter 3, ask during the next class.

THE END