Computer Programming & Problem Solving
CS100

Mrs Sanga G. Chaki
Department of Computer Science and Engineering
National Institute of Technology, Goa
November, 2022

What is C?

1. A general purpose programming language
2. Used in operating systems, device drivers, protocol stacks, computer architectures of super computers to microcontrollers
3. Designed by: Dennis Ritchie almost 50 years ago
4. Successor to language B

Why do we need C?

1. Computers do not understand English, they understand Binary
2. Humans are more comfortable with natural languages
3. Programming languages bridge this gap
4. All languages have some common features

How does any Language work?

1. Language
    a) Character-set
    b) Symbols
    c) Logic
    d) Syntax
    e) Grammar

Example code

```c
// Online C compiler to run C program online
#include <stdio.h>

int main() {
    int num1, num2, sum;
    printf("Please enter 2 integer numbers: ");
    scanf("%d %d", &num1, &num2);
    sum = num1 + num2;
    printf("The sum = %d", sum);
    return 0;
}
```

General Rules to write C code

1. Each instruction in a C program is written as a separate statement.
2. The statements in a program must appear in the same order in which we wish them to be executed
3. No unnecessary blank spaces are allowed
4. All statements are entered in small case letters
5. Every C statement must end with a ;.
6. Comment about the program should be enclosed within /* */ or //.
7. main( ) is a collective name given to a set of statements. This name has to be main( ), it cannot be anything else. All statements that belong to main( ) are enclosed within a pair of braces { }

General Rules to write C code

1. main( ) is a function. Every function has a pair of parentheses ( ) associated with it. printf() and scanf() are also functions.
2. Any variable used in the program must be declared before using it.
3. The general form of printf( ) function is,
   a) printf ("<format string>", <list of variables>) ;
   b) <format string> can contain, %f for printing real values, %d for printing integer values
4. The general form of scanf( ) function is,
   a) scanf ("<format string>", <address of variables>) ;
   b) To specify address we use the & operator

General Rules to write C code

1. #include<stdio.h>
   a) Stdio.h → header file. It contains some important code already written
   b) #include → Instruction to add this file to our code.
   c) Math.h → Math header file, for pow(), sqrt().

Types of C instructions: Important Aspects of C

1. How it represents and stores data
2. How it operates upon this data
3. How it accomplishes input and output
4. How it lets you control the sequence of execution of instructions in a program.

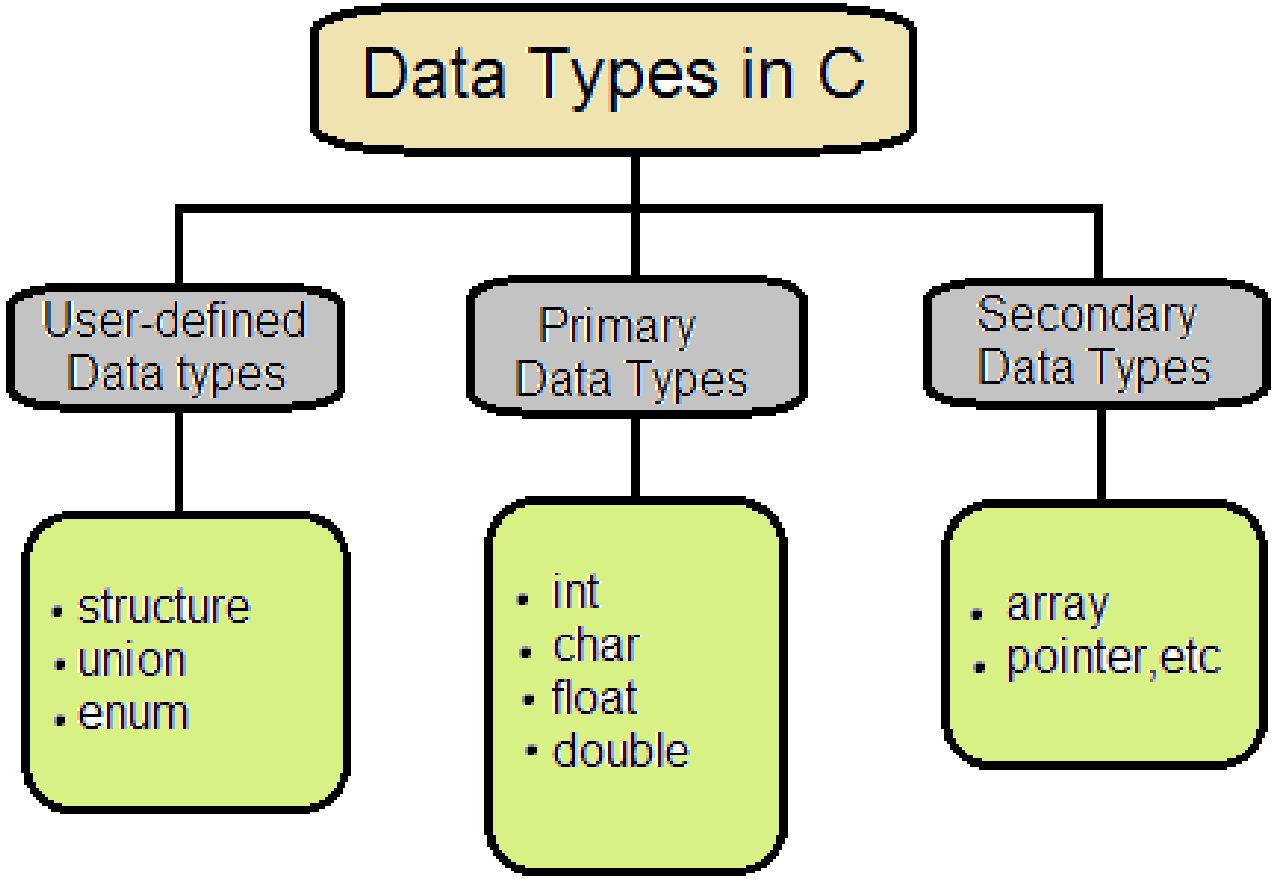
How C represents and stores data

Represent Data: C character set and operators

1. A character denotes any alphabet, digit or special symbol used to represent information.
2. Using these characters we create meaningful data.

Alphabets	A, B, ….., Y, Z a, b, ……, y, z
Digits	0, 1, 2, 3, 4, 5, 6, 7, 8, 9
Special symbols	~ ‘ ! @ # % ^ & * ( ) _ - + = | \ { } [ ] : ; " ' < > , . ? /

Represent Data: C Data Types



1. A particular kind of data item:
   a) Defines the values it can take
   b) The operations that can be performed on it.

C Constants and Variables

1. A constant is an entity that doesn’t change
2. A variable is an entity that may change
3. They can be of any datatype described in the previous slide.

C Variables – Rules to construct Variable names

1. A variable name: combination of 1-31 alphabets, digits or underscores
2. Do not create unnecessarily long variable names
3. Create problem related meaningful variable names
4. The first character must be an alphabet or underscore.
5. No commas or blanks are allowed.
6. No special symbol other than an underscore can be used
7. To distinguish between the variable names it is compulsory to declare the type of any variable

C Keywords

1. Keywords are the words whose meaning has already been explained to the C compiler.
2. The keywords cannot be used as variable names
3. There are only 32 keywords available in C.

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

How C operates upon data

Type Declaration Instruction

1. To declare the type of variables used in a C program
2. Any variable used in the program must be declared before using it in any statement.
3. The type declaration statement is written at the beginning of main( ) function.

Arithmetic Instructions - Rules

1. A C arithmetic instruction consists of a variable name on the left hand side of =
2. And variable names & constants on the right hand side of =.
3. The variables and constants appearing on the right hand side of = are connected by arithmetic operators like +, -, *, and /.
4. That's why we cannot write
   a) num1+num2=sum; → Invalid
5. Execution of an arithmetic statement:
   a) First, the right hand side is evaluated using constants and the numerical values stored in the variable names.
   b) This value is then assigned to the variable on the left-hand side.

Arithmetic Instructions - Rules

1. C allows only one variable on left-hand side of =.
2. No operator is assumed to be present.
3. There is no operator for performing exponentiation operation – use pow()
4. Here pow( ) function is a standard library function.
5. #include <math.h> is used here to ensure that the pow( ) function works correctly

Arithmetic Instructions

A C arithmetic statement can be of three types

1. Integer mode arithmetic statement - This is an arithmetic statement in which all operands are either integer variables or integer constants.
    a) Rule: An arithmetic operation between an integer and integer always yields an integer result
2. Real mode arithmetic statement - This is an arithmetic statement in which all operands are either real constants or real variables.
    a) Rules: An operation between a real and real always yields a real result.
3. Mixed mode arithmetic statement - This is an arithmetic statement in which some of the operands are integers and some of the operands are real
    a) Rules: An operation between an integer and real always yields a real result.