Flow Chart
and
C Variables and Operations

A. Sahu and P. Mitra

Dept of Comp. Sc. & Engg.

Indian Institute of Technology Guwahati

Outline

- Quick Recap of Number System
- Flow Charts and Problem solving
- C Programming: Variable, data type and operations

Computing in this course

- Given a Problem :
  - In English description
- Solve using Computer
  - Design methods to solve the problem
  - **Analyze the designed solution for correctness**
  - Design flow chart to for the design method to solve the problem, Write Pseudocode
  - /source code
  - Compile and rule the code : **You may get some error**
  - Test the code (with some Input): **you get some error**

Computing in this course

- Given a Problem :
  - In English description
- Solve using Computer
  - Design methods to solve the problem
  - **Analyze the designed solution for correctness**
  - Design flow chart to for the design method to solve the problem, Write Pseudocode

Computing

Programming

  - Write source code
  - Compile and run the code : **You may get some error**
  - Test the code (with some Input): **you may get some error**

Problem Solving Example 1

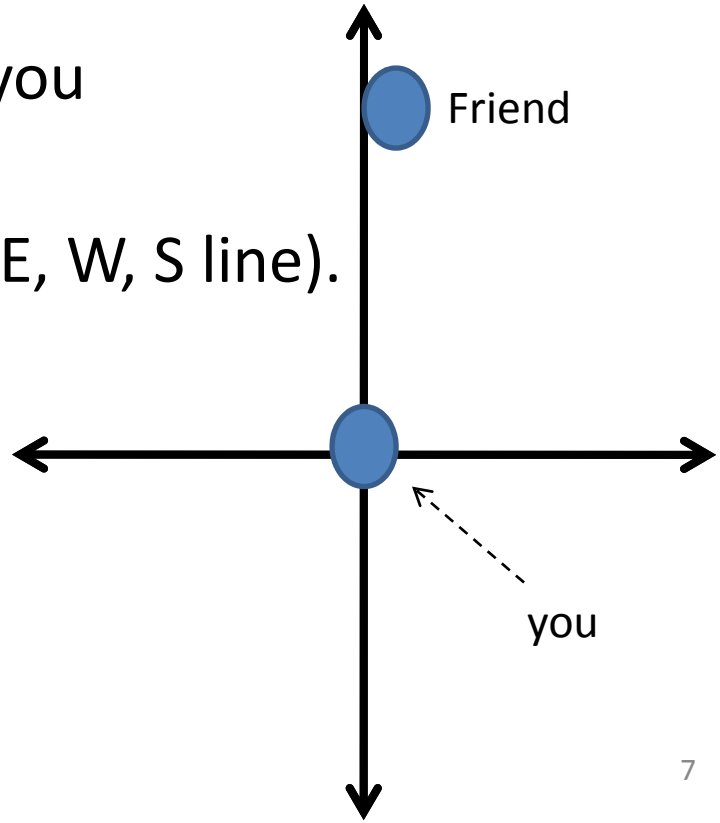
- Given a positive integer, is this number divisible by 3?
- Method 1
  - Divide the number by three and test for the reminder
    - Repeated substation of 3 till you get the result less than 3
    - Divide using usual method and get reminder

Problem Solving Example 1

- Given a positive integer, is this number divisible by 3?
- Method 2
  - Sum all the digits of the number and test the divisibility of Sum by three, repeat the same.
  - If sum is divisible by three then number is divisible by three. How you got to know?
  - Some one must have **solved this problem and proved the correctness** of the solution.
    - ➡ Solution will work for all the cases

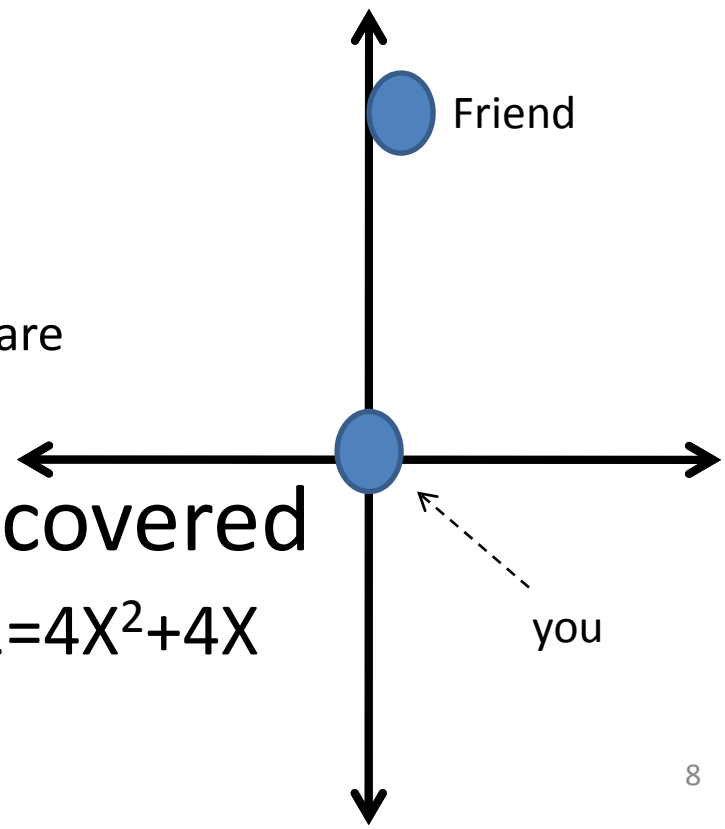
Problem Solving Example 2

- Problem: Searching for your friend
  - Your friend is near to square and you are exactly at the square
  - **You don't know which direction and how much distance from the square**
  - You need to find your friend, you can see only unit distance
  - Your friend in on the lines (N, E, W, S line).
- Any Solution?
  - How you will search him?
  - Any approach
  - Think for 2-3 minutes



Problem Solving Example 2

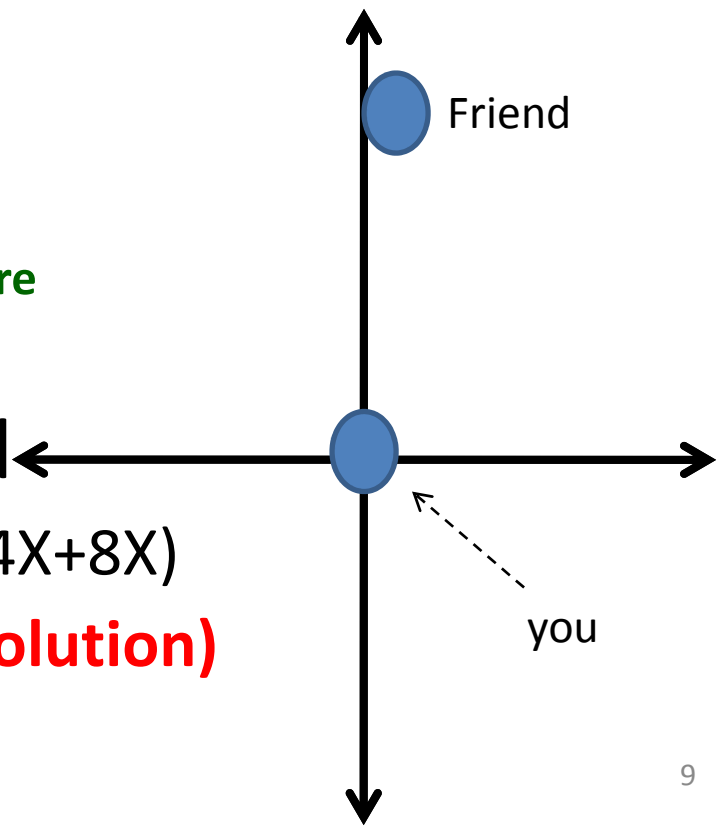
- Problem: Searching for your friend
  - **You don't know which direction and how much distance from the square**
- Solution?
  - K=1;
  - While(not found)
    - For all directions
      - Go K meter and return to square
    - K=k+1;
- Analysis :Total distance covered
  - $4*2(1+2+..+X)=8*X*(X+1)/2=4X^2+4X$



Problem Solving Example 2

- Problem: Searching for your friend
  - **You don't know which direction and how much distance from the square**
- Better Solution? **YES**
  - K=1;
  - While(not found)
    - **Go in Next directions**
      - **Go K meter and return to square**
    - **K=k*2;**
- Analysis: distance covered
  - At max $(1+2+2^2+2^3+..+X+2X+4X+8X)$

    $< 16*X < \mathbf{4X^2+4X}$ **(earlier solution)**



Computing in this course

- Given a Problem :
  - In English description
- Solve using Computer
  - Design methods to solve the problem
  - **Analyze the designed solution for correctness**
  - Design flow chart to for the design method to solve the problem
  - Write Pseudocode/source code
  - Compile and rule the code : **You may get some error**
  - Test the code (with some Input): **you get some error**

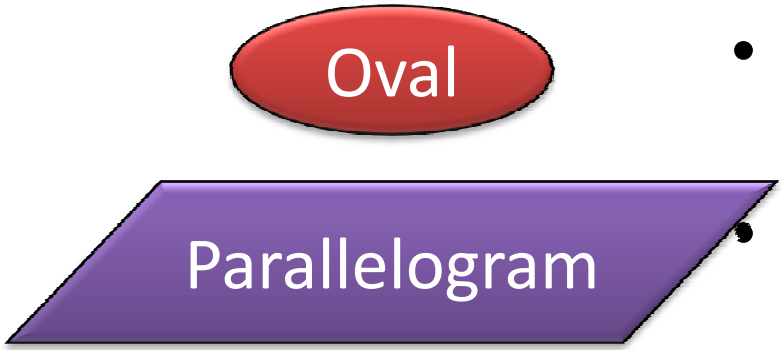
Computing

Programming

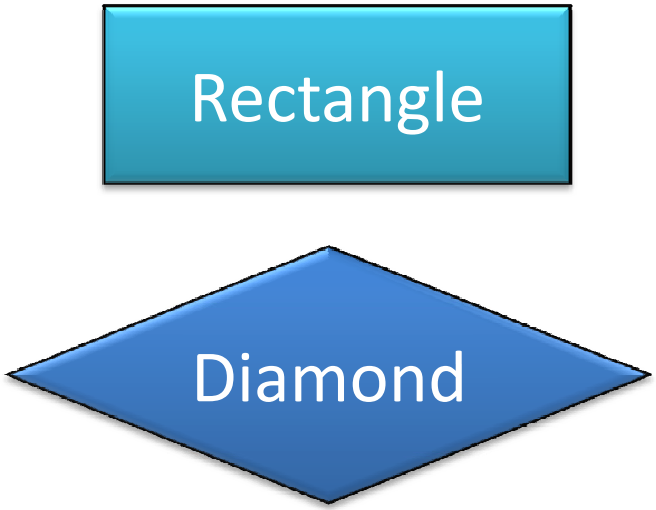
The Flowchart (quick recap)

- Flowchart shows logic of an algorithm
- Emphasizes individual steps
- Their interconnections e.g. control flow from one action to the next

Flowchart Symbols (quick recap)



- **Oval:** begin and end
- **Parallelogram:** Input or Output



- **Rectangle:** process to be carried out
  - add/sub/div/inc
- **Diamond:** Decision/Branch to be made IF/THEN/ELSE

- **Flow line:** Direction of logic flow in program

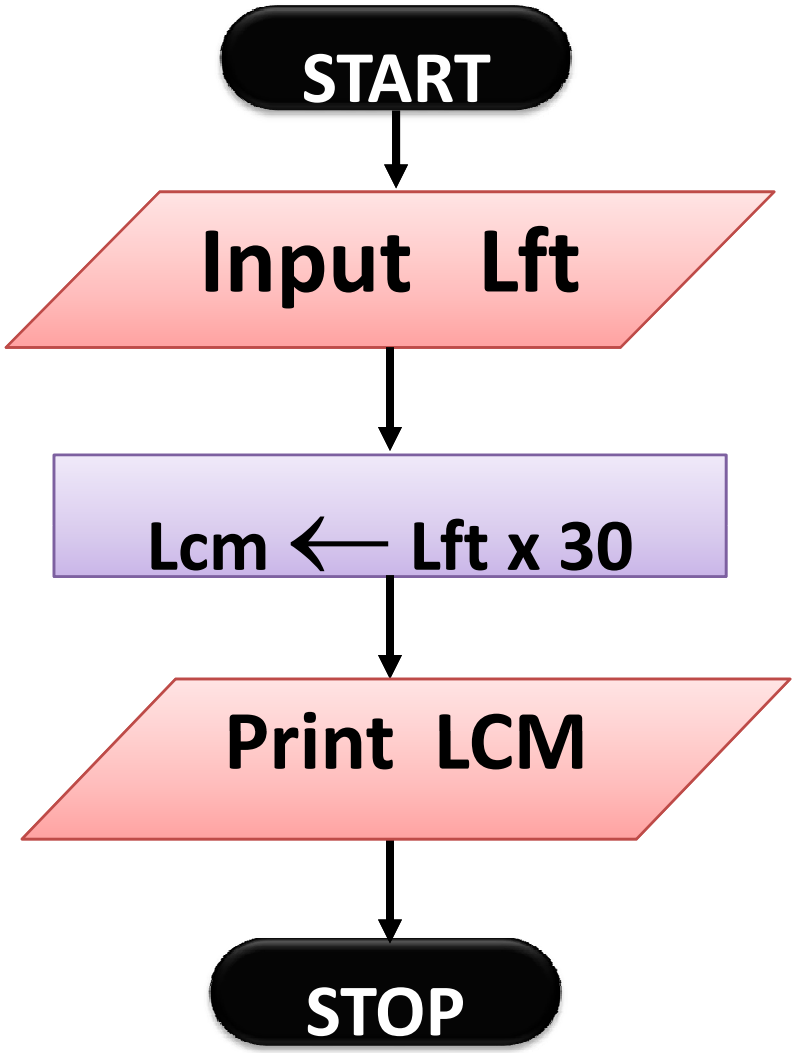
Example 1: Length FT to Length CM (quick recap)

- Draw a flowchart to convert the length in feet to centimeter.
- Steps:
  - *Input the length in feet (LFT)*
  - *Calculate the length in cm (LCM) by multiplying LFT with 30*
  - *Print length in cm (LCM)*

Example 1: Length FT to Length CM (quick recap)

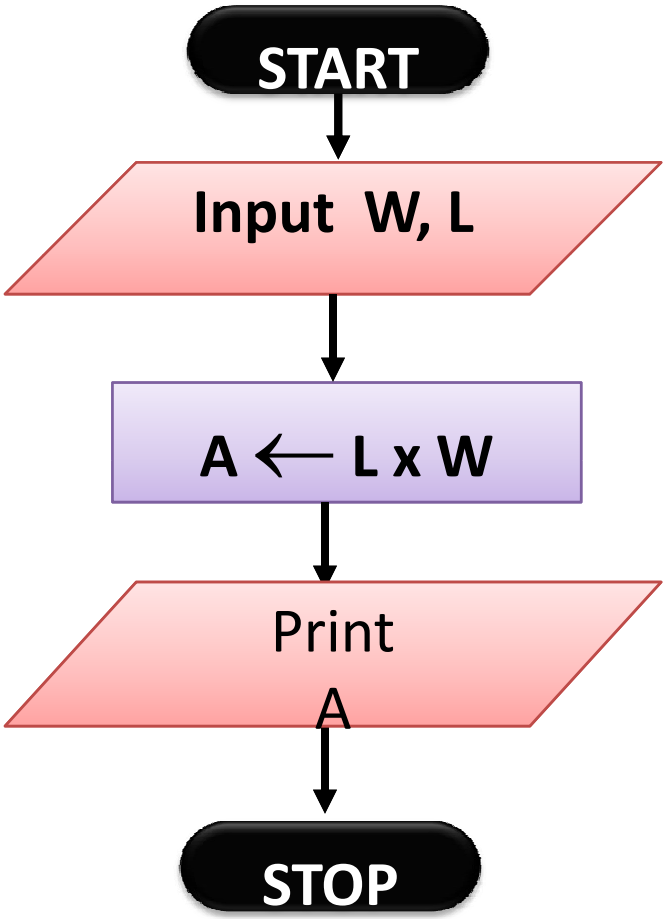
- Step 1: Input Lft
- Step 2: Lcm ← Lft x 30
- Step 3: Print Lcm

Flowchart



Example 2: Area of Rectangle (quick recap)

Read the two sides of a rectangle and calculate its area.



- Step 1: Input W,L
- Step 2: A ← L x W
- Step 3: Print A

Example 3 : Roots of a Quadratic Equation (quick recap)

- Quadratic equation

$$ax^2+bx+c=0$$

- Calculate $d = \text{sqrt}\ (\ b^2-4*a*c\ )$
- Roots are:

$$\boldsymbol{x1} = (-b + d)/2a$$

$$\boldsymbol{x2} = (-b - d)/2a$$

Example 3 : Roots of a Quadratic Equation

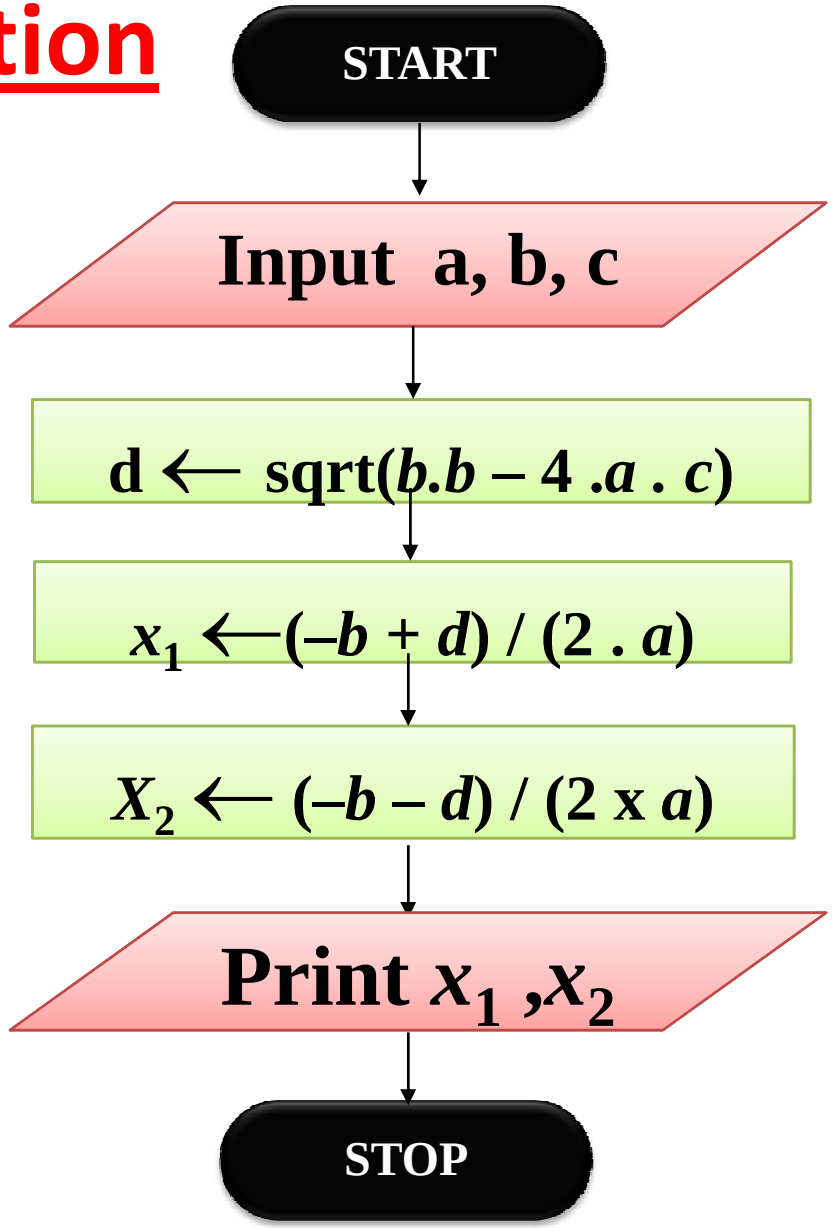
- Step 1:

  Input a, b, c
- Step 2: calculate d

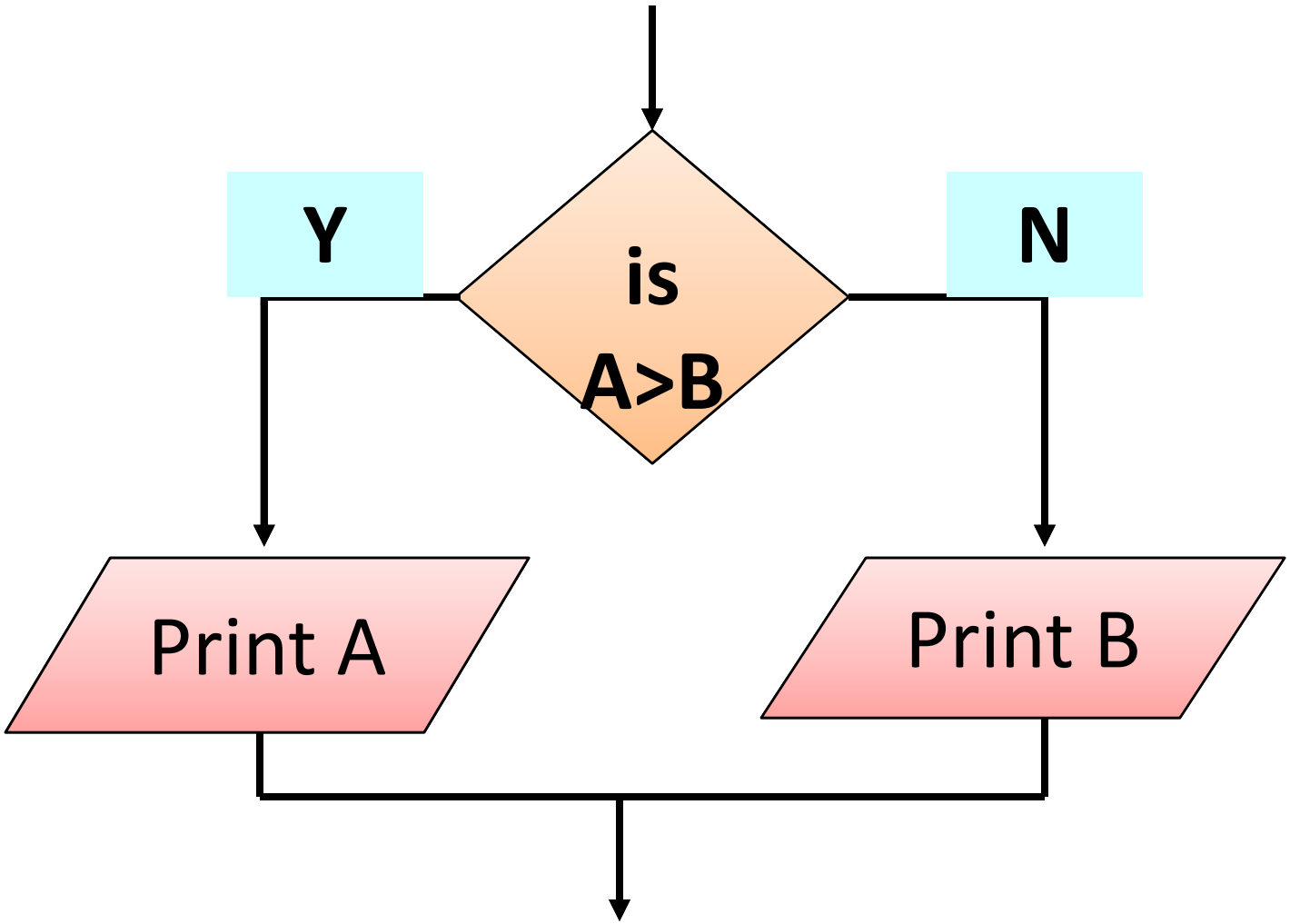
  $d \leftarrow$ sqrt (bxb-4xaxc)
- Step 3: calculate x1

  $x1 \leftarrow (-b + d) / (2 \text{ x } a)$
- Step 4: calculate x2

  $x2 \leftarrow (-b - d) / (2 \text{ x } a)$
- Step 5: Print $x1$, $x2$

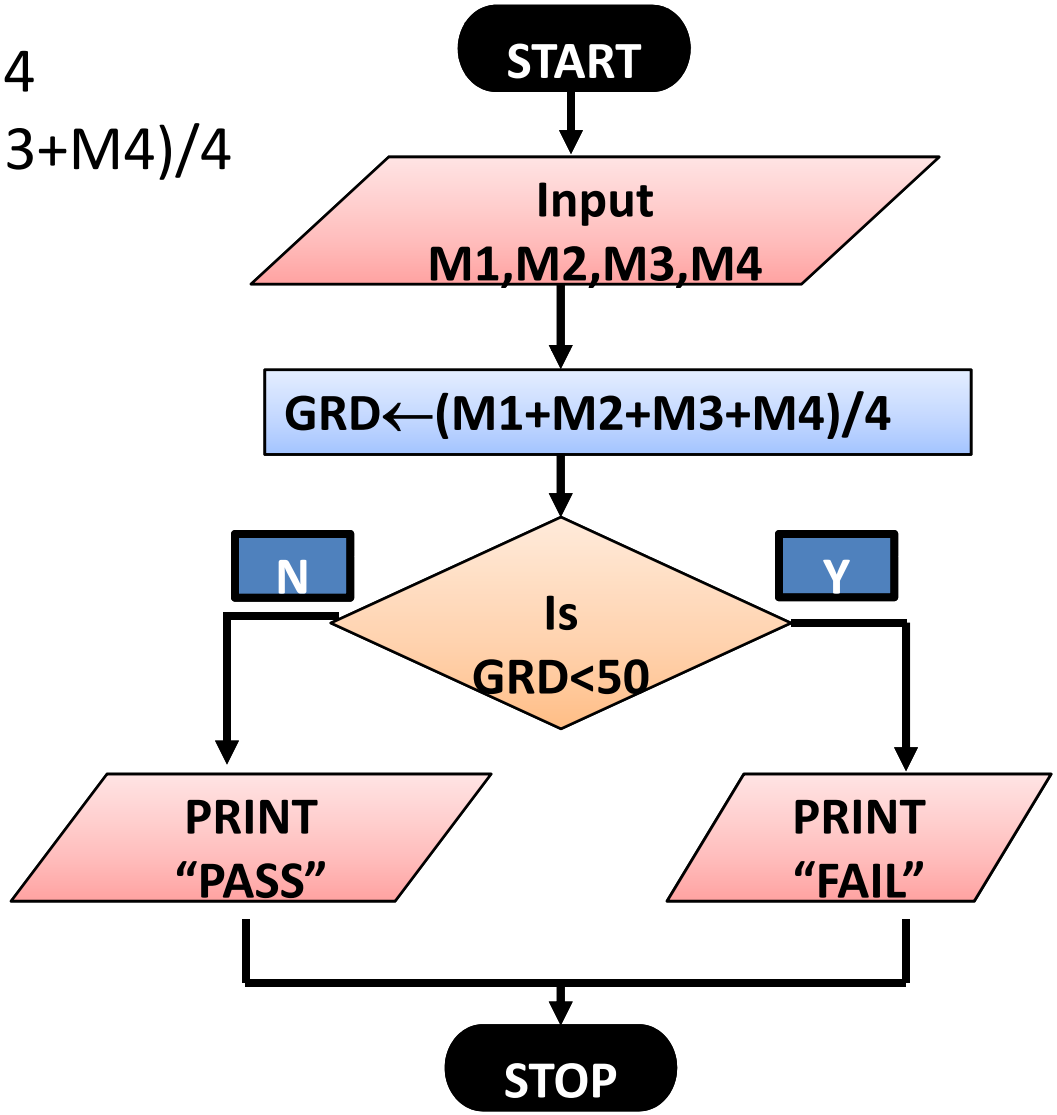


Decision Structure



Example 4 : Grade Calculation

Step 1: Input M1,M2,M3,M4
Step 2: GRD ← (M1+M2+M3+M4)/4
Step 3: if (GRD <50) then
Print "FAIL"
else
Print "PASS"
endif

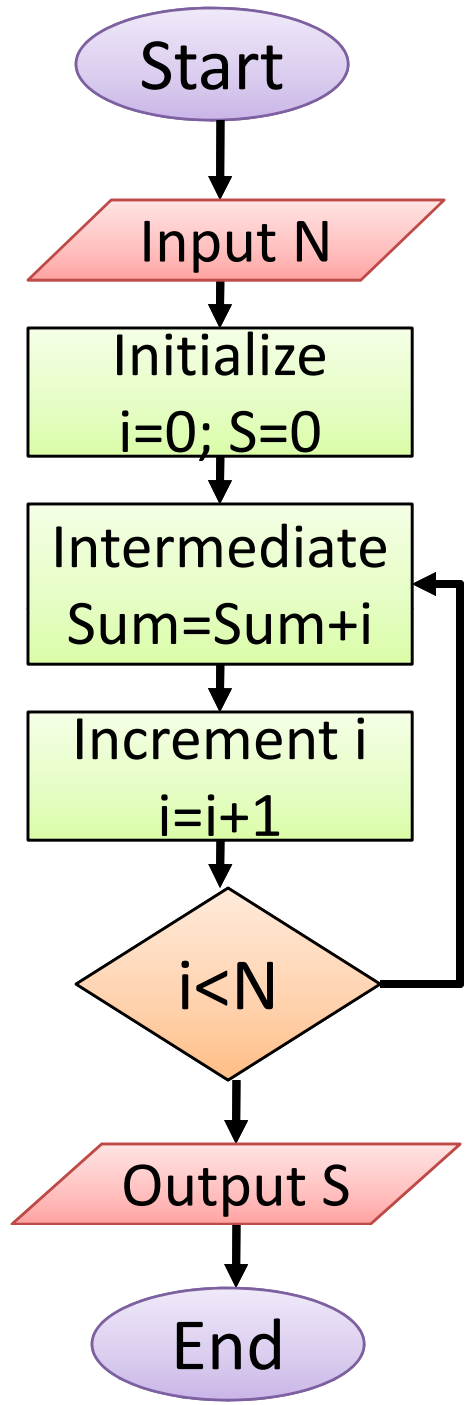


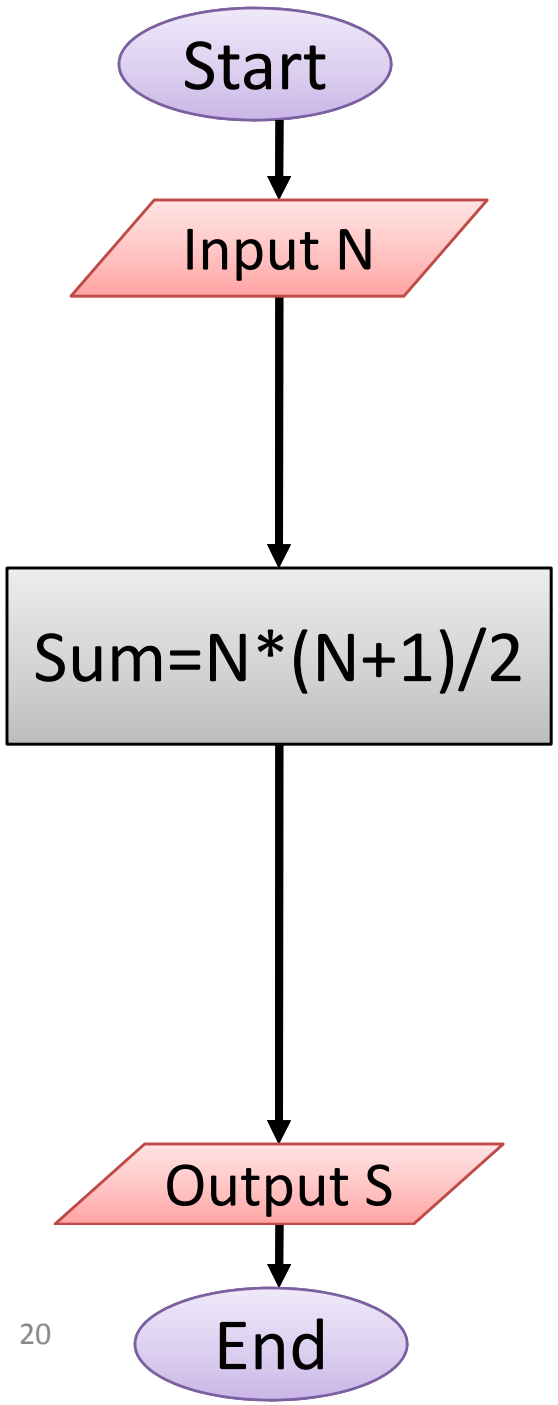
Flow Chart Example 4

- Sum of first N natural number
- Two methods
  - Method 1

    Sum= 1+2+3+…+N
  - Method 2

    Sum = N(N+1)/2





C Programming
Basic Example, Types and Operators

Basic C Programming

- Variable Name and Declaration
- Data types and sizes
- Constants
- Declarations of variable
- Operators in C
- Examples : Basic operations

C Programming : Sum of A and B

```
#include <stdio.h>

int main(){
 int A,B, S;
 printf("Enter two
          numbers ");
 scanf("%d %d",&A,&B);

 S=A+B;

 printf("Res=%d", S);
 return 0;
}
```

Header file:
Standard Input/Output

Variable Declarations

Printing message

Asking for inputs

Compute

Output Result

Identifier in C : Name

- Is a unique name
  - That simply references to memory locations, which can hold values (data).
- Identifiers give unique names
  - To various objects in a program
  - To **variable and function**
- Are formed by
  - Combining letters (both upper and lowercase)
  - Digits (0–9)
  - Underscore (_).

Rules for naming Identifier in C

- First character of an identifier
  - Must be a letter (non-digit) including underscore (_).
- Space: Blank or white space character is not permitted Space, Tab
  - linefeed, carriage-return, form-feed, vertical-tab, and newline characters
- Length
  - Can be any length but implementation dependent
- **Reserved words/keywords cannot be used.**

Identifier in C : Variable Name Examples

Correct	Wrong
secondName	2ndName /* starts with a digit */
_addNumber	%calculateSum /* contains invalid character */
charAndNum	char /* reserved word */

Identifier in C : Variable Name Examples

Correct	Wrong
annual_rate	annual rate /* contains a space */
stage4mark	my\nName /* contains new line character, \n */

What Are Variables in C?

- **Variables** in C have the same meaning as variables in algebra. That is, they represent some unknown, or variable, value.

$$x = a + b$$

$$z + 2 = 3(y - 5)$$

- Remember that variables in algebra are represented by a single alphabetic character.

Variables in C

- Variables are
  - Named blocks of memory
  - Valid identifier.
- Variable have two properties in syntax:
  - Name — a unique identifier
  - Type — what kind of value is stored.
- It is identifier, that
  - Value may change during the program execution.
- Every variable stored in the computer's memory
  - Has a name, a value and a type.

C Variable : Example

Correct
`int x, y, z, my_data = 4;;`
`short number_one;`
`long TypeofCar;`
`unsigned int positive_number;`
`char Title;`
`float commission, yield = 4.52;`
`char the_initial = 'M'; //A char`
`char studentName[20]=“India“;//A string`

C Variable : Example

Wrong	Comments
`int 3a, 1, -p;`	Start with digit or -
`short number+one;`	Have +
`long #number;`	Start wit #

Variable Naming Conventions

- C programmers generally agree on the following **conventions** for naming variables.
  - Begin variable names with lowercase letters
  - Use meaningful identifiers
  - Separate "words" within identifiers with underscores or mixed upper and lower case.
  - Examples: `surfaceArea`, `surface_Area`, `surface_area`
  - **Be consistent!**

Case Sensitivity

- C is **case sensitive**
  - It matters whether an **identifier**, such as a variable name, is uppercase or lowercase.
  - Example:

    area

    Area

    AREA

    ArEa

  are all seen as different variables by the compiler.

Which Are Legal Identifiers?

AREA	area_under_the_curve
3D	num45
Last-Chance	#values
x_yt3	pi
num$	%done
lucky***	

Which Are Legal Identifiers?

AREA

3D

Last-Chance

x_yt3

num$

lucky***

area_under_the_curve

num45

#values

pi

%done

Naming Conventions

- Use all uppercase for **symbolic constants** (used in **#define** preprocessor directives).
- Examples:

```
#define PI 3.14159
#define AGE  52
```

C Programming : Sum of A and B

```
#include <stdio.h>

int main(){
 int A,B, S;
 printf("Enter two
          numbers ");
 scanf("%d %d",&A,&B);

 S=A+B;

 printf("Res=%d", S);
 return 0;
}
```

Header file:
Standard Input/Output

Variable Declarations

Printing message

Asking for inputs

Compute

Output Result

C Keywords/Reserve Words

- Reserved words are : not available for redefinition
- Can not be used as variable name
- Have special meaning in C

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

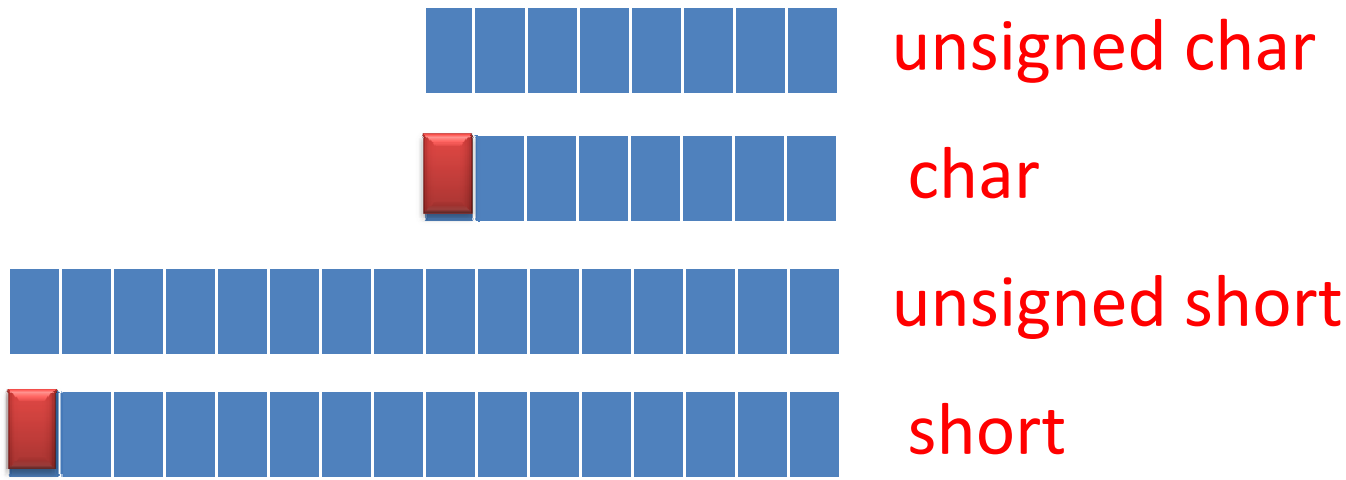
Data type and Size

- In C, data type categorized as:
  - Primitive Types in ANSI C (C89)/ISO C (C90) –
    - **`char`, `short`, `int`, `float` and `double`.**
  - Primitive Types added to ISO C (C99) –
    - **`long int, long double`**
  - User Defined Types
    - **`struct`, `union`, `enum` and `typedef` (will be discussed later).**
  - Derived Types
    - **`pointer, array` and `function pointer` (will be discussed later).**

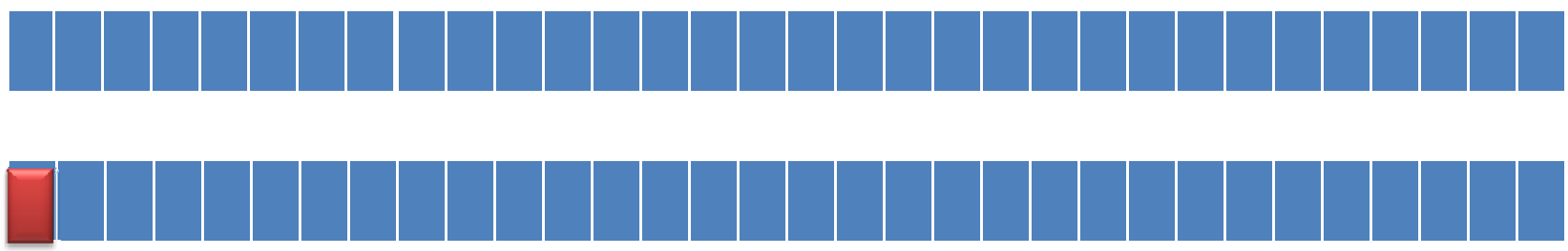
Numeric Data Type

- **char, short, int, long int**
  - char : 8 bit number (1 byte=1B)
  - short: 16 bit number (2 byte)
  - int : 32 bit number (4B)
  - long int : 64 bit number (8B)
- **float, double, long double**
  - float : 32 bit number (4B)
  - double : 64 bit number (8B)
  - long double : 128 bit number (16B)

Numeric Data Type



Unsigned int



int

Testing size of Numeric Data

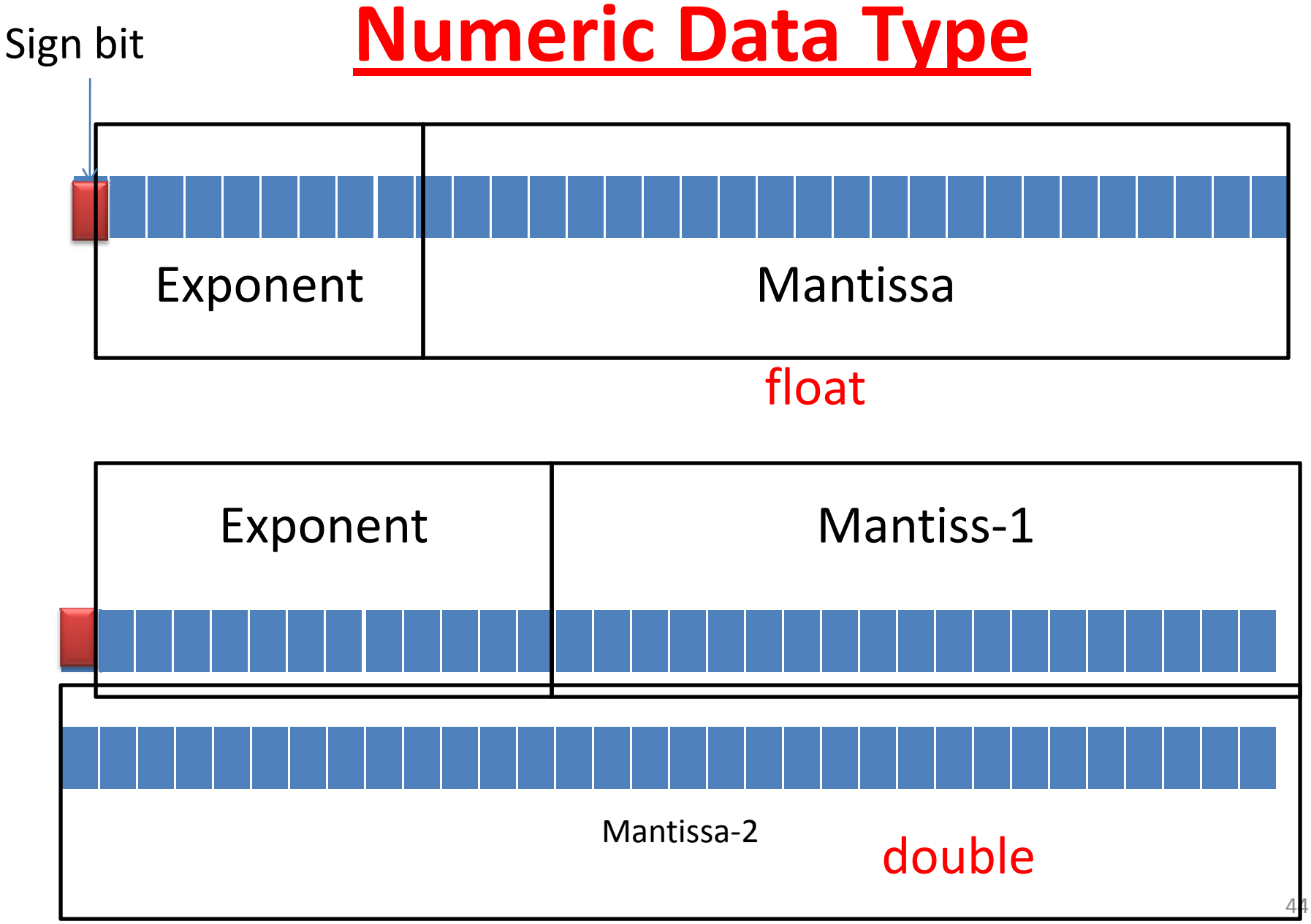
```
#include<stdio.h>
int main(){
  printf("size of char %d\n", sizeof(char)); //1
  printf("size of short %d\n",sizeof(short)); //2
  printf("size of int %d\n",sizeof(int)); //4
  printf("size of long int %d\n",sizeof(long int)); //8

  printf("size of float \n",sizeof(float)); //4
  printf("size of double %d\n",sizeof(double));//8
  printf("size of long double  %d\n",
                                  sizeof(long double));//16
  return 0;
}
```

Numeric Data Type

- **char, short, int, long int**
  - We have : Signed and unsigned version
  - char (8 bit)
    - char : -128 to 127, we have +0 and -0 ☺ ☺ Fun
    - unsigned char: 0 to 255
  - int : -2^{31} to $2^{31}-1$
  - unsigned int : 0 to $2^{32}-1$
- **float, double, long double**
  - For fractional, real number data
  - All these numbered are signed and get stored in different format

Numeric Data Type



Thanks