MAHIDOL UNIVERSITY

Wisdom of the Land

[SCPY204]

Computer Programing

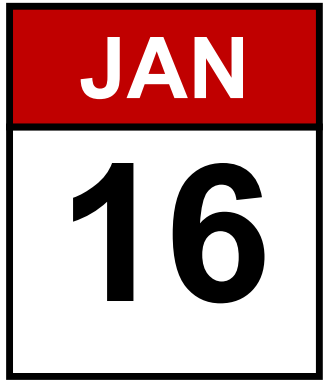
for Physicists

Class 02: 16 Jan 2023

Content: Data, Data type, program control, condition and loop, function and recursion, variable and scope

Instructor: Puwis Amatyakul

2023



Review

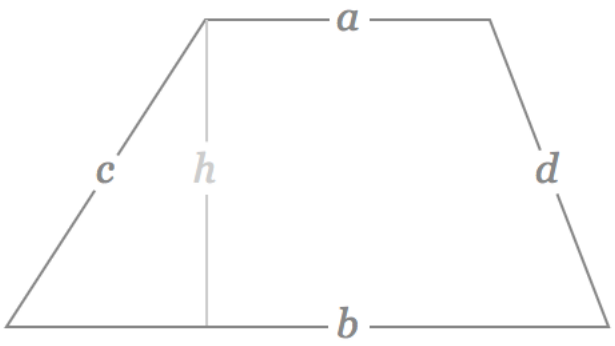
Steps in Programming

1. Problem analysis
2. Planning and design
3. Coding
4. Testing/debugging
5. Documentation

Steps in Programming

Problem: Write a program to calculate trapezoid.

1. Problem analysis



$$A = \frac{a+b}{2}h$$

- How to calculate area of trapezoid?
- Variables involved?
- Program need inputs.
- Do the calculation.

Steps in Programming

Problem: Write a program to calculate trapezoid.

2. Planning and design

Pseudo code

Pseudocode is an **informal** high-level description of the operating principle of a computer program or other algorithm.

Flow chart

Flowchart is a type of diagram that represents an algorithm, workflow or process, showing the steps as boxes of various kinds, and their order by connecting them with arrows.

Steps in Programming

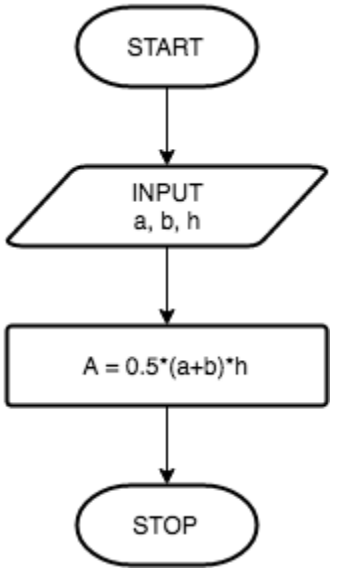
Problem: Write a program to calculate trapezoid.

2. Planning and design

Pseudo code

```
START
READ a
READ b
READ h
COMPUTE A = 0.5*(a+b)*h
PRINT A
STOP
```

Flow chart



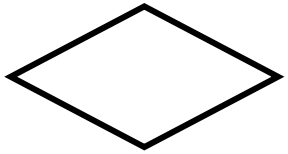
Try: www.draw.io

Standard Flow Chart Symbol

Terminator: START/STOP

Input/Output

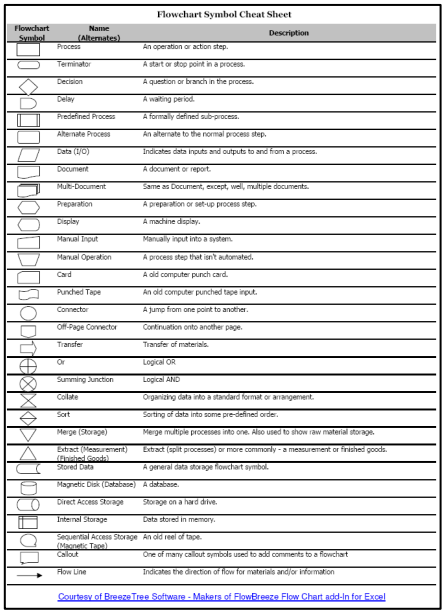
Process



Decision/Compare

Subprogram

Flow



Flowchart Symbol Cheat Sheet

Flowchart Symbol	Name (Alternates)	Description
	Process	An operation or action step.
	Terminator	A start or stop point in a process.
	Decision	A question or branch in the process.
	Delay	A waiting period.
	Predefined Process	A formally defined sub-process.
	Alternate Process	An alternate to the normal process step.
	Data (I/O)	Indicates data inputs and outputs to and from a process.
	Document	A document or report.
	Multi-Document	Same as Document, except, well, multiple documents.
	Preparation	A preparation or set-up process step.
	Display	A machine display.
	Manual Input	Manually input into a system.
	Manual Operation	A process step that isn't automated.
	Card	A old computer punch card.
	Punched Tape	An old computer punched tape input.
	Connector	A jump from one point to another.
	Off-Page Connector	Continuation onto another page.
	Transfer	Transfer of materials.
	Or	Logical OR
	Summing Junction	Logical AND
	Collate	Organizing data into a standard format or arrangement.
	Sort	Sorting of data into some pre-defined order.
	Merge (Storage)	Merge multiple processes into one. Also used to show raw material storage.
	Extract (Measurement) (Finished Goods)	Extract (split processes) or more commonly - a measurement or finished goods.
	Stored Data	A general data storage flowchart symbol.
	Magnetic Disk (Database)	A database.
	Direct Access Storage	Storage on a hard drive.
	Internal Storage	Data stored in memory.
	Sequential Access Storage (Magnetic Tape)	An old reel of tape.
	Callout	One of many callout symbols used to add comments to a flowchart
	Flow Line	Indicates the direction of flow for materials and/or information

Courtesy of BreezeTree Software - Makers of FlowBreeze Flow Chart add-in for Excel

More on: http://www.breezetree.com/images/flow-chart-symbols.png

Steps in Programming

Problem: Write a program to calculate trapezoid.

3. **Coding**

C?	Python?	Matlab?

4. Testing/debugging
5. Documentation

Today's Goals

Part I: Data – Data type

Part II: Program control, condition and loop

Part III: Function and recursion

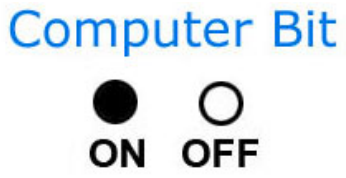
Part IV: Variable and scope

We are going to talk about **Data**!

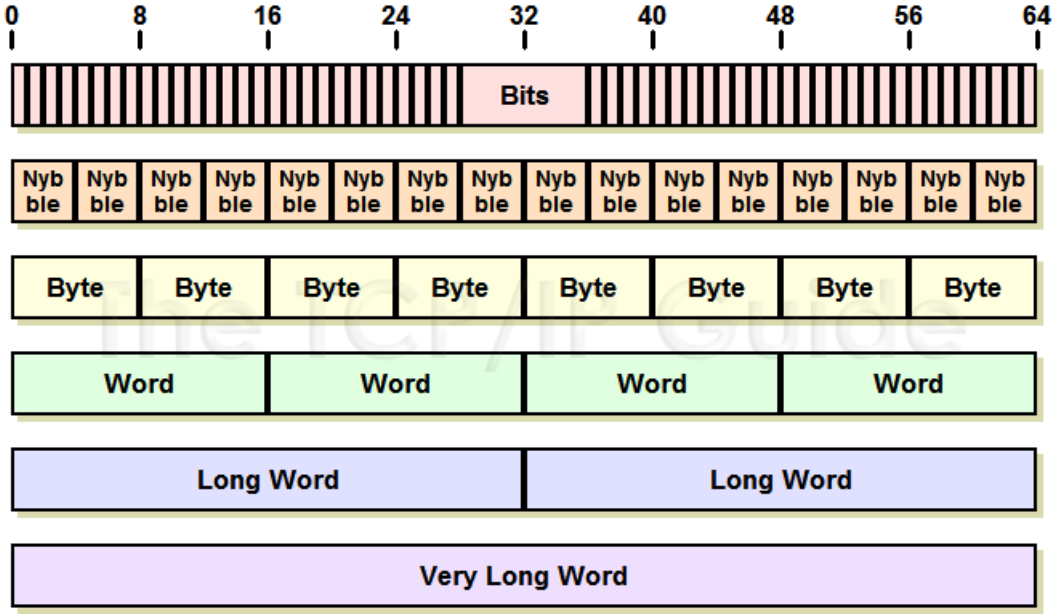
a little **Bit**...

[**B**inary Digi**t**]

Bit and Byte



Numbering system:
- Binary
- Decimal
- Hexadecimal



Prefixes for multiples of bits (bit) or bytes (B)

Decimal			Binary		
Value	SI		Value	IEC	JEDEC
1000	k	kilo	1024	Ki kibi	K kilo
1000^2	M	mega	1024^2	Mi mebi	M mega
1000^3	G	giga	1024^3	Gi gibi	G giga
1000^4	T	tera	1024^4	Ti tebi	–
1000^5	P	peta	1024^5	Pi pebi	–
1000^6	E	exa	1024^6	Ei exbi	–
1000^7	Z	zetta	1024^7	Zi zebi	–
1000^8	Y	yotta	1024^8	Yi yobi	–

V • T • E

Data types

This week: C first!

Data types in C

1.Fundamental Data Types

1. Integer types
2. Floating type
3. Character type

2.Derived Data Types

1. Arrays
2. Pointers
3. Structures
4. Enumeration

Variable Type	Keyword	Bytes Required	Range	Format
Character (signed)	Char	1	-128 to +127	%c
Integer (signed)	Int	2	-32768 to +32767	%d
Float (signed)	Float	4	-3.4e38 to +3.4e38	%f
Double	Double	8	-1.7e308 to + 1.7e308	%lf
Long integer (signed)	Long	4	2,147,483,648 to 2,147,438,647	%ld
Character (unsigned)	Unsigned char	1	0 to 255	%c
Integer (unsigned)	Unsigned int	2	0 to 65535	%u
Unsigned long integer	unsigned long	4	0 to 4,294,967,295	%lu
Long double	Long double	10	-1.7e932 to +1.7e932	%Lf

ASCII TABLE

Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char	Decimal	Hex	Char
0	0	[NULL]	32	20	[SPACE]	64	40	@	96	60	`
1	1	[START OF HEADING]	33	21	!	65	41	A	97	61	a
2	2	[START OF TEXT]	34	22	"	66	42	B	98	62	b
3	3	[END OF TEXT]	35	23	#	67	43	C	99	63	c
4	4	[END OF TRANSMISSION]	36	24	$	68	44	D	100	64	d
5	5	[ENQUIRY]	37	25	%	69	45	E	101	65	e
6	6	[ACKNOWLEDGE]	38	26	&	70	46	F	102	66	f
7	7	[BELL]	39	27	'	71	47	G	103	67	g
8	8	[BACKSPACE]	40	28	(	72	48	H	104	68	h
9	9	[HORIZONTAL TAB]	41	29	)	73	49	I	105	69	i
10	A	[LINE FEED]	42	2A	*	74	4A	J	106	6A	j
11	B	[VERTICAL TAB]	43	2B	+	75	4B	K	107	6B	k
12	C	[FORM FEED]	44	2C	,	76	4C	L	108	6C	l
13	D	[CARRIAGE RETURN]	45	2D	-	77	4D	M	109	6D	m
14	E	[SHIFT OUT]	46	2E	.	78	4E	N	110	6E	n
15	F	[SHIFT IN]	47	2F	/	79	4F	O	111	6F	o
16	10	[DATA LINK ESCAPE]	48	30	0	80	50	P	112	70	p
17	11	[DEVICE CONTROL 1]	49	31	1	81	51	Q	113	71	q
18	12	[DEVICE CONTROL 2]	50	32	2	82	52	R	114	72	r
19	13	[DEVICE CONTROL 3]	51	33	3	83	53	S	115	73	s
20	14	[DEVICE CONTROL 4]	52	34	4	84	54	T	116	74	t
21	15	[NEGATIVE ACKNOWLEDGE]	53	35	5	85	55	U	117	75	u
22	16	[SYNCHRONOUS IDLE]	54	36	6	86	56	V	118	76	v
23	17	[ENG OF TRANS. BLOCK]	55	37	7	87	57	W	119	77	w
24	18	[CANCEL]	56	38	8	88	58	X	120	78	x
25	19	[END OF MEDIUM]	57	39	9	89	59	Y	121	79	y
26	1A	[SUBSTITUTE]	58	3A	:	90	5A	Z	122	7A	z
27	1B	[ESCAPE]	59	3B	;	91	5B	[	123	7B	{
28	1C	[FILE SEPARATOR]	60	3C	<	92	5C	\	124	7C	|
29	1D	[GROUP SEPARATOR]	61	3D	=	93	5D	]	125	7D	}
30	1E	[RECORD SEPARATOR]	62	3E	>	94	5E	^	126	7E	~
31	1F	[UNIT SEPARATOR]	63	3F	?	95	5F	_	127	7F	[DEL]

Source: https://commons.wikimedia.org/wiki/File:ASCII-Table-wide.svg


THE MARTIAN

QUIZ time!

Today's Goals

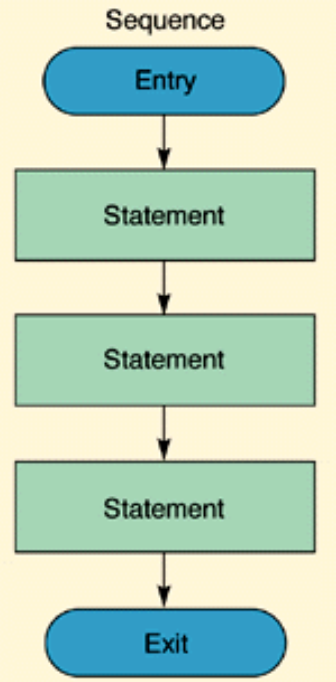
Part I: Data – Data type

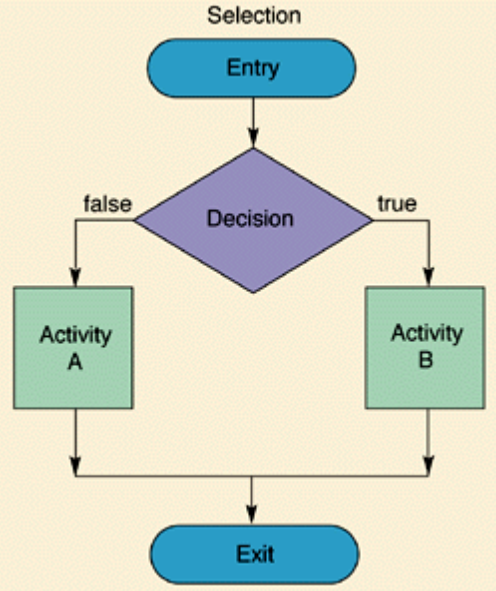
Part II: Program control, condition and loop

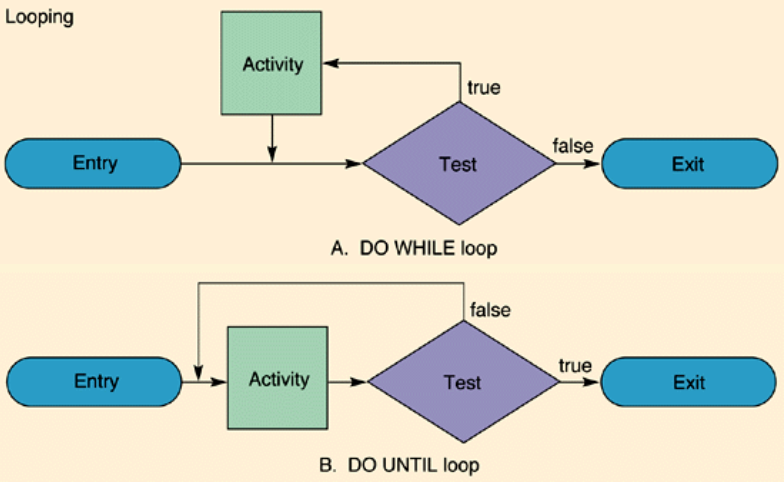
Part III: Function and recursion

Idea

Part II: Program control, condition and loop (and their nested)







C Basic: Syntax

C Basic: Variables

Type	Description
char	Typically a single octet(one byte). This is an integer type.
int	The most natural size of integer for the machine.
float	A single-precision floating point value.
double	A double-precision floating point value.
void	Represents the absence of type.

Variable Definition in C

type variable_list;

```
int    i, j, k;
char   c, ch;
float  f, salary;
double d;
```

C Basic: Variables

```c
#include <stdio.h>

int main () {

   /* variable definition: */
   int a, b;
   int c;
   float f;

   /* actual initialization */
   a = 10;
   b = 20;

   c = a + b;
   printf("value of c : %d \n", c);

   f = 70.0/3.0;
   printf("value of f : %f \n", f);

   return 0;
}
```

C Basic: Operator

Arithmetic Operators

Operator	Description	Example
+	Adds two operands.	A + B = 30
−	Subtracts second operand from the first.	A − B = -10
*	Multiplies both operands.	A * B = 200
/	Divides numerator by de-numerator.	B / A = 2
%	Modulus Operator and remainder of after an integer division.	B % A = 0
++	Increment operator increases the integer value by one.	A++ = 11
--	Decrement operator decreases the integer value by one.	A-- = 9

C Basic: Operator

Relational Operators

Operator	Description	Example
==	Checks if the values of two operands are equal or not. If yes, then the condition becomes true.	(A == B) is not true.
!=	Checks if the values of two operands are equal or not. If the values are not equal, then the condition becomes true.	(A != B) is true.
>	Checks if the value of left operand is greater than the value of right operand. If yes, then the condition becomes true.	(A > B) is not true.
<	Checks if the value of left operand is less than the value of right operand. If yes, then the condition becomes true.	(A < B) is true.
>=	Checks if the value of left operand is greater than or equal to the value of right operand. If yes, then the condition becomes true.	(A >= B) is not true.
<=	Checks if the value of left operand is less than or equal to the value of right operand. If yes, then the condition becomes true.	(A <= B) is true.

C Basic: Operator

Logical Operators

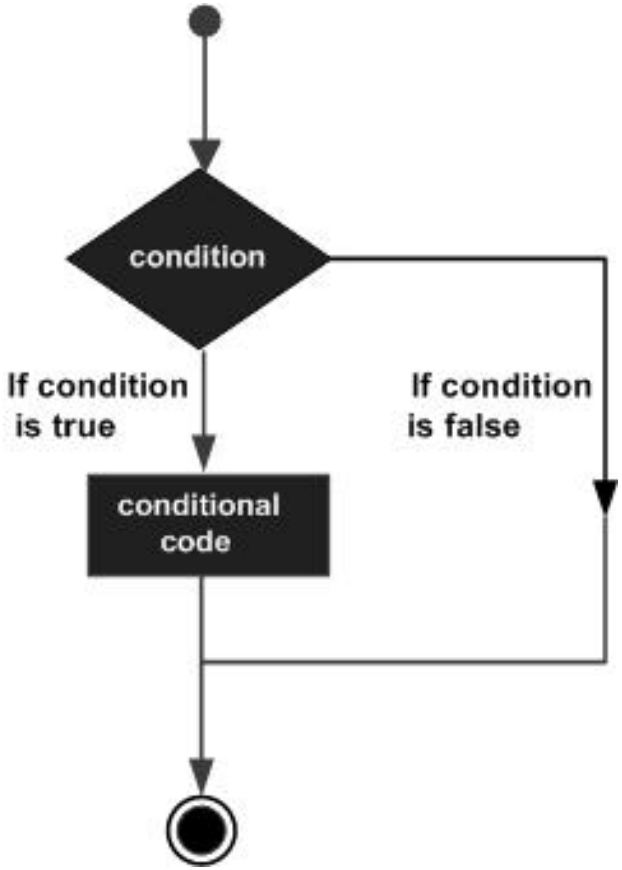
Operator	Description	Example
&&	Called Logical AND operator. If both the operands are non-zero, then the condition becomes true.	(A && B) is false.
||	Called Logical OR Operator. If any of the two operands is non-zero, then the condition becomes true.	(A || B) is true.
!	Called Logical NOT Operator. It is used to reverse the logical state of its operand. If a condition is true, then Logical NOT operator will make it false.	!(A && B) is true.

Steps in Programming

Problem: Write a program to calculate trapezoid.

Can you now do this in C?

C Basic: Decision Making (condition)

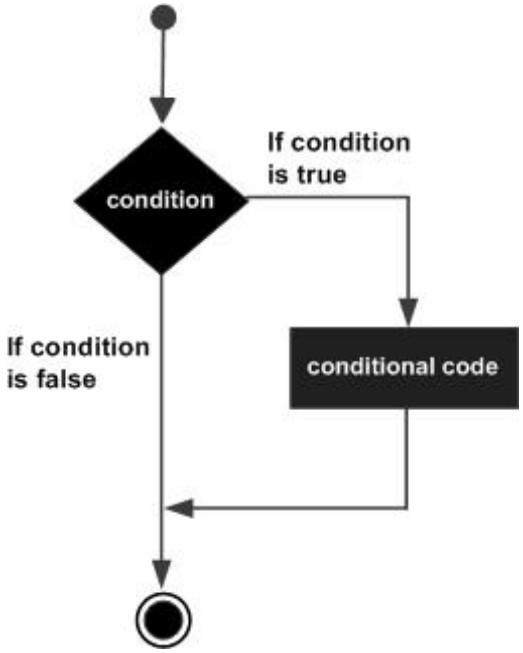


S.N.	Statement & Description
1	**if statement** An **if statement** consists of a boolean expression followed by one or more statements.
2	**if...else statement** An **if statement** can be followed by an optional **else statement**, which executes when the Boolean expression is false.
3	**nested if statements** You can use one **if** or **else if** statement inside another **if** or **else if** statement(s).
4	**switch statement** A **switch** statement allows a variable to be tested for equality against a list of values.
5	**nested switch statements** You can use one **switch** statement inside another **switch** statement(s).

C Basic: Decision Making (condition)

If statement

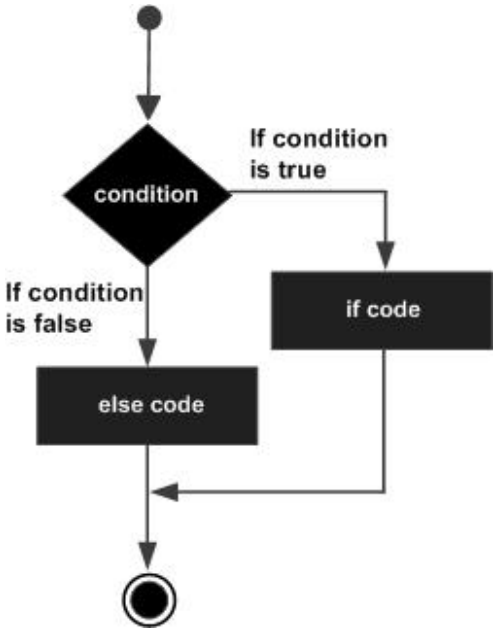
```c
#include <stdio.h>

int main () {

   /* local variable definition */
   int a = 10;

   /* check the boolean condition using if statement */

   if( a < 20 ) {
      /* if condition is true then print the following */
      printf("a is less than 20\n" );
   }

   printf("value of a is : %d\n", a);

   return 0;
}
```



C Basic: Decision Making (condition)

If … else statement

```c
#include <stdio.h>

int main () {

   /* local variable definition */
   int a = 100;

   /* check the boolean condition */
   if( a < 20 ) {
      /* if condition is true then print the
following */
      printf("a is less than 20\n" );
   }
   else {
      /* if condition is false then print the
following */
      printf("a is not less than 20\n" );
   }

   printf("value of a is : %d\n", a);

   return 0;
}
```



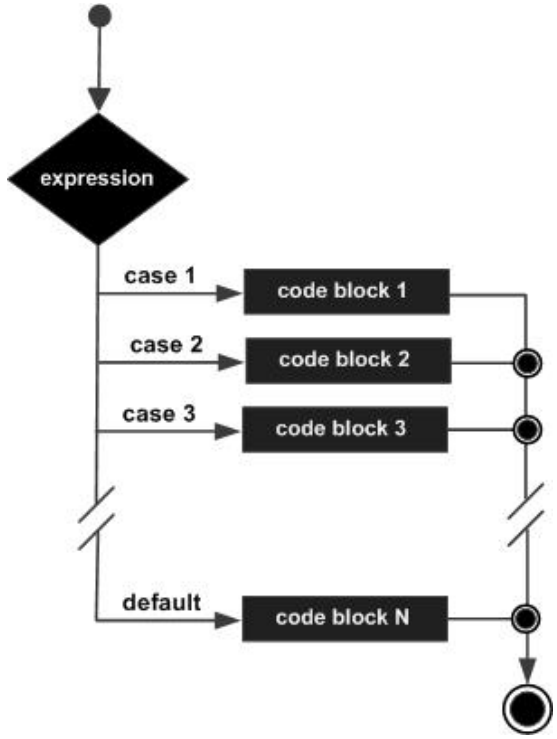
C Basic: Decision Making (condition)

Nested If statement

```c
#include <stdio.h>

int main () {

   /* local variable definition */
   int a = 100;
   int b = 200;

   /* check the boolean condition */
   if( a == 100 ) {

      /* if condition is true then check the following */
      if( b == 200 ) {
         /* if condition is true then print the following */
         printf("Value of a is 100 and b is 200\n" );
      }
   }

   printf("Exact value of a is : %d\n", a );
   printf("Exact value of b is : %d\n", b );

   return 0;
}
```

C Basic: Decision Making (condition)

```c
#include <stdio.h>

int main () {

   /* local variable definition */
   char grade = 'B';

   switch(grade) {
     case 'A' :
        printf("Excellent!\n" );
        break;
     case 'B' :
     case 'C' :
        printf("Well done\n" );
        break;
     case 'D' :
        printf("You passed\n" );
        break;
     case 'F' :
         printf("Better try again\n" );
        break;
     default :
         printf("Invalid grade\n" );
   }

   printf("Your grade is  %c\n", grade );

   return 0;
}
```



Switch statement

Exercise

Part II: **Program control, condition** and loop

EX: C Program to Check Whether a Number is Positive, Negative or Zero

```c
#include <stdio.h>

int main() {
    int number;
    /*
     * Take a number as input from user
     */
    printf("Enter a Number\n");
    scanf("%d", &number);

    if(number > 0) {
        printf("%d is Positive Number", number);
    } else if (number < 0) {
        printf("%d is Negative Number", number);
    } else {
        printf("Input Number is Zero");
    }

    return 0;
}
```

Exercise

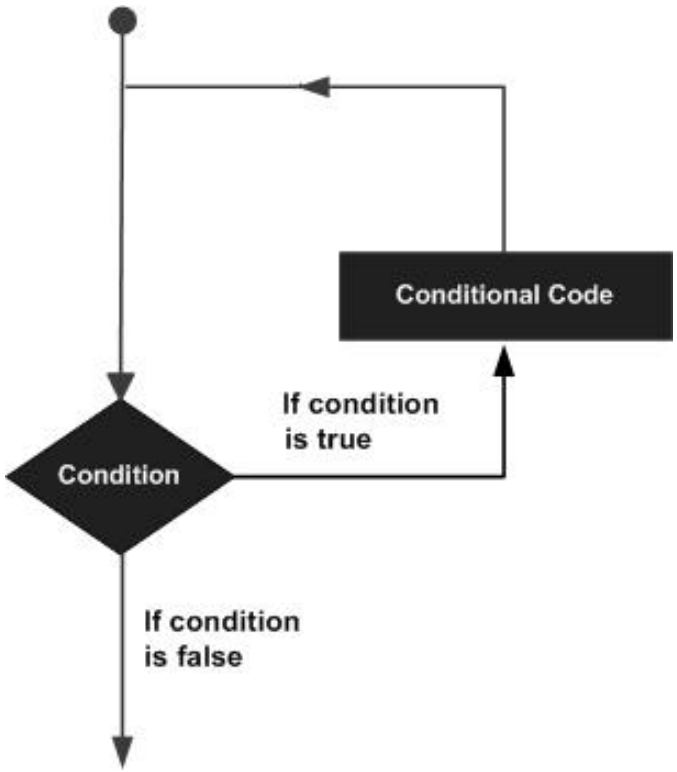
Part II: **Program control, condition** and loop

EX: C program to check a number is Even of Odd using

1. If statement
2. switch case statement

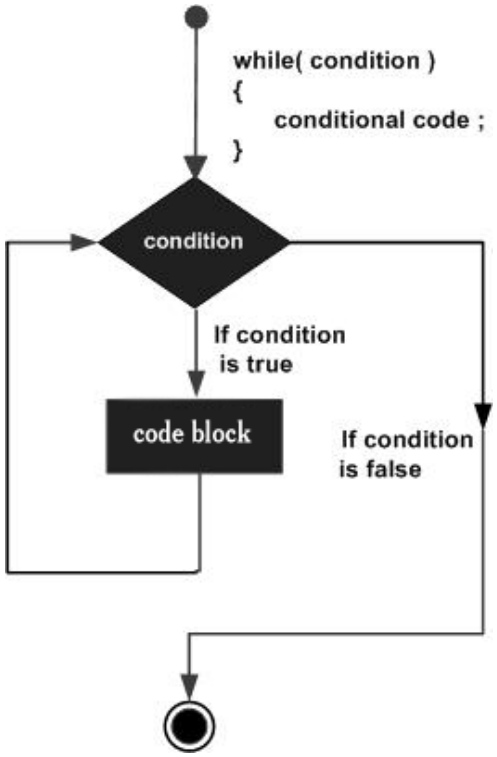
Save to ex01.c

C Basic: Loop



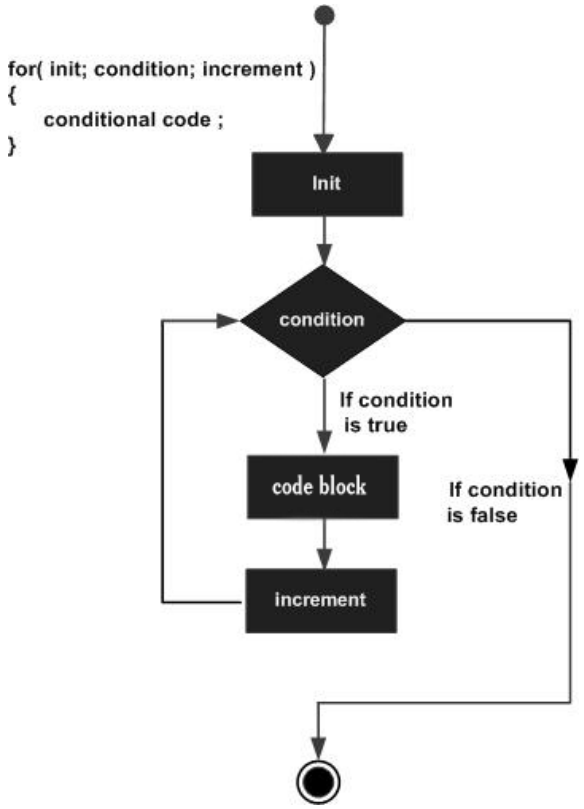
S.N.	Loop Type & Description
1	**while loop** Repeats a statement or group of statements while a given condition is true. It tests the condition before executing the loop body.
2	**for loop** Executes a sequence of statements multiple times and abbreviates the code that manages the loop variable.
3	**do...while loop** It is more like a while statement, except that it tests the condition at the end of the loop body.
4	**nested loops** You can use one or more loops inside any other while, for, or do..while loop.

Basic C: Loop (while)




```c
#include <stdio.h>

int main () {

   /* local variable definition */
   int a = 10;

   /* while loop execution */
   while( a < 20 ) {
      printf("value of a: %d\n", a);
      a++;
   }

   return 0;
}
```

Basic C: Loop (while)




```c
#include <stdio.h>

int main () {

   int a;

   /* for loop execution */
   for( a = 10; a < 20; a = a + 1 ){
      printf("value of a: %d\n", a);
   }

   return 0;
}
```

Exercise

Part II: **Program control, condition** and loop

EX: Find summation from 1 to 100

Save to ex02a.c

EX: Find summation of even number from 1 to 100

Save to ex02b.c

For all exercise ex01a,b and ex02a,b,
- Zip and send to puwis.ama@mahidol.ac.th -