16 Monday — January 2017

Week-3

016-349

08.00
09.00
10.00
11.00
12.00
01.00
02.00
03.00
04.00
05.00
06.00

2017 — January

S	M	T	W	T	F	S
1	2	3	4	5	6	7
8	9	10	11	12	13	14
15	16	17	18	19	20	21
22	23	24	25	26	27	28
29	30	31				

January 2017 — Tuesday 17

017-348

Week-3

C Programming

Basic - Advanced

08.00
- Vs code / codeblocks / k-develop → code editor (write & edit code)
  - like notebook
09.00
→ download vs code for windows
10.00
→ download gcc for windows (MINGW) → ☰ → Apply changes
→ Environment variable
11.00
· Mingw/bin (path copy) → control panel → system → Advanced system settings → Environment variable → System variables.
12.00
→ path → Edit → New → path copy → ok → ok → ok
01.00
→ Our laptop :-
02.00
Settings → System → About → Advanced system settings → Environment variable → system variable → path → Edit → New → path paste → ok → ok
03.00
→ open vs code → new folder (C_programs) → new file (Hello_world.c)
04.00
1st program ⟹ SUCCESS
05.00
Hello world.
In terminal
gcc Hello_world.c
./a.exe
06.00

→ download C/C++ Extension pack.

2017 — February

S	M	T	W	T	F	S
		1	2	3	4	5
6	7	8	9	10	11	12
13	14	15	16	17	18	19
20	21	22	23	24	25	26
27	28					

Compiler : rules of C are defined there

IDE - Integrated development Environment.

18 Wednesday — January 2017

1" Program :- Hello-world.C

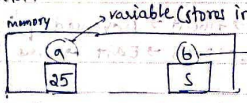
.C - cpgm extension

```
#include <stdio.h>
void main ().
{
printf ("Hello world");
}
```

Terminal
gcc Hello-world.C
./a.exe (windows)

CHAPTER-1

Variables :- Name of the memory location which stores some data.



Syntax :- data-type variable-name ;

```
int   number = 25 ;
char  star = '*' ;
float pi   = 3.14 ;
```

Rules

(i) Variables are Case-sensitive int A=30; and int a=40; are diff
(ii) Alpabet or underscore (start) int _age=20; ✓ int 1age=18; X
(iii) special characters are not allowed. (Except underscore)
(iv) No comma or blank space
(v) do not start with number

Variables should be meaningful

1970 → old language

January 2017 — Thursday 19

Data-types

These are the special keywords that tells which type of data will be stored in particular variable.

Data-type	Format specifier	Size in byte
char	%c	1
int	%d	2 or 4 (int & longint)
float	%f	4
double	%lf	8

Eg:- int age = 22;
float pi = 3.14 ;
char hashtag = '#' ;

Signed char ⇒ -128 to 127
Unsigned char ⇒ 0 to 255

char = ' '

Constants — Values are fixed
— Values that don't change

int constants 1, 2, 3, 0, -1, -2 etc
real constants 1.0, 2.0, 3.14, -24 etc
char constants 'a', 'b', 'A', '#' etc

Keywords

reserved words that have special meaning to the Complier

32 keywords in C

Eg:- int, float, for, while, return Etc.
auto - to declare local variables.

Program structure

```
// Documentation                    used to include the libraries which are required in our program.
#include<stdio.h>                   # preprocessor directive
int main ( ) {                      # main function. - start of program.
    printf("Hello World");          # body of the program
    return 0 ;                      # ; - terminator of line
}                                   # zero errors.
```

Comments:-

Lines that are not part of program

```
//  - single line
/*  - } Multiple-line comments
*/
```

Output :

```
printf ("Hello world");

print ("Anything\n");
```

\n - new line Escape Character

Cases (Output)

(i) integer

```
printf (" age is %d ", age);
```

(ii) real numbers - float

```
printf (" The value of pi is %f", pi);
```

(iii) Character

```
printf (" star looks like %c", star);
```

Format specifiers - %d, %f, %c
(%d - double int, %f - float, %c - char)

Input

```
scanf ("%d", &age);
```
(& - address)

2. Program:- Sum of two numbers.

```
int a,b;
printf (" Enter value of a & b: ");
scanf ("%d%d", &a, &b);
printf ("sum of %d & %d is : %d ", a, b, a+b);
```

Compilation:-
A computer program that translates C code into machine code (0/1) if C code does not have any errors.

Hello.C → C complier → a.exe (window)
a.out (mac or linux)

Practice Qs 1 : Area of a square (side is given).
Area = a * a

Practice Qs 2 : Area of a Circle (radius is given)

Area = 3.14 * r * r. ⇒ πr^2

H.W set
(i) perimeter of rectangle 2(l+b) ⇒ 2*(l+b)
(ii) take n from user & o/p its Cube (n*n*n)
(iii) write comments for programs a & b

Extra
(i) simple interest SI = (P*T*R)/100
(ii) compound interest CI = p*[1+r/100]^t − p → #math.h> pow(-,-)
(iii) F° − C° C° = (°F − 32) × 5/9 → 100°C → 212°F
(iv) °C − °F °F = (°C × 9/5) + 32
(v) ~~perimeter of circle~~ ⇒ 2πr
(vi) perimeter of square ⇒ 4*a
(vii) area of rectangle ⇒ l*b

CHAPTER - 2 Instructions and Operators.

Instructions:
These are statements in a programs.

Types
(i) type declaration instruction
(ii) Arithmetic instruction
(iii) Control instruction.

(i) type declaration Instruction

Declare var before using it.

valid
```
int a = 22;
int b = a;
int c = b+1;
int d=5, e;
```

invalid
```
int b = a+2; #Error
int a = 2;
```

declaration multiple variable.
```
int a,b,c;
a=b=c=1;
```
a=1
b=1
c=1

invalid
```
int a=b=c=4; //Error
```

```
int a,b,c=5;
```
only

25 Wednesday — January 2017

(ii) arithmatic Instruction

a + b

a: operand 1; b: operand 2

+: operator (symbol which tells which operation will be performed)

NOTE - Single variable on the LHS

valid	invalid
a=b+c ;	b+c = a
a=b*c ;	a=bc
a=b|c ;	a=b^c //No Error but not (power)

NOTE - pow(x,y) for x to the power y

^ - Bitwise operator (XOR gate)

Modular Operator — / - quotient

returns remainder of int

3%2 = 1

-3%2 = -1

% - not works for float type

2017 January

S M T W T F S S M T W T F S
1 2 3 4 5 6 7 8 9 10 11 12 13 14
15 16 17 18 19 20 21 22 23 24 25 26 27 28
29 30 31

January 2017 — Thursday 26

Type Conversion.

int op int → int

int op float → float (A)

float op float → float

int op float → float

small container, big container → big container (required)

decimals not required	decimals required
a = 3/2 // 1	a = 3.0/2 // 1.500000
a = 2/3 // 0	a = 2.0/3 // 0.666666

Practice Qs 3:

solve: int a = 1.999999;

Type conversion

int a = (int) 1.999999;

printf ("%d", a); #1

Implicit: compiler do for itself

Explicit: User have to tell to make conversion

implicit

float b = 1; float ← int

but

int c = 1.2; int ← float (X)

so explicit

2017 February

S M T W T F S S M T W T F S
1 2 3 4 5 6 7 8 9 10 11
12 13 14 15 16 17 18 19 20 21 22 23 24 25
26 27 28

Operator precedence

BODMAS

↓ parentheses
PODMAS

*, /, %

↓

+, −

↓

= (assignment operator)

X = 4+9*10 // 94

X = 4*3/6*2 // 4

Associativity (for same precedence)

Left to right

X = 4*3/6*2 ⇒ 4//

Practice Qs-4

① 5*2 − 2*3 // 4
② 5*2 / 2*3 // 15
③ 5*(2/2)*3 // 15
④ 5+2/2*3 // 8

Instructions (Control instructions)

Used to determine flow of program.

ⓐ Sequence Control
ⓑ Decision Control → if-else
ⓒ loop Control → for, while
ⓓ Case Control → switch

Operators :

ⓐ Arithmatic operator :- +, −, *, /, %
ⓑ relational operator :- ==, !=, >, >=, <, <=
ⓒ logical operator :- && (AND) (if-else), || (OR), ! (NOT)
ⓓ bitwise operator :—
ⓔ Assignment operator :—
ⓕ Ternery operator :—

ⓒ logical operator

AND			OR			NOT	
1	1	1	1	1	1	0	1
1	0	0	1	0	1	1	0
0	1	0	0	1	1		
0	0	0	0	0	0		

30 Monday — January 2017

Week 5

Precedence

1	!	
2	*, /, %	} Arithmatic
3	+, -	
4	<, <=, >, >=	} relational
5	==, !=	
6	&&	} logical
7	||	
8	=	

Assignment operator :-

=, +=, -=, *=, /=, %=

a=b

a = a+2 or a+=2

Practice Q: 5

(a) Write a program to check number is divisible by 2 or not

even or odd

```
n % 2 == 0
```

P.Q: 6

(a) even or odd

February 2017 — Thursday 02

Week 5

P.Qs-7

Valid or not

(i) int a = 8^8 — Valid (Bit wise 8 XOR 8 = 0)

(ii) int x ; int y = x ; — Valid

(iii) int x , y = x ; — invalid (error)

(iv) char class = 'xx' ; — invalid (warning)

char stores single character

P.Qs-8

print 1(true) & 0(false) for following statements.

(a) if it's sunday & its snowing → True

(b) if its monday or it's raining → True

(c) if a number is greaterthan 9 & lessthan 100 → True.

2 digit number

(a)
```
int isSunday = 1;
int isSnowing = 1;
printf("%d", isSunday && isSnowing);   // 1 (AND operator)
```

(b)
```
int isMonday = 0;
int isRaining = 1;
printf("%d", isMonday || isRaining);   // 1 (OR operator)
```

(c)
```
int x = 25;
printf("%d", (x>9) && (x<100));   // 1
```

HW set

(a) print the average of 3 numbers.

(b) to check if given character is digit or not.

(c) smallest number

(d) isDigit() function

<ctype.h>

```
char ch;
scanf("%c", &ch);
printf("%d", ch>='0' && ch<='9')
```

ASCII

'0' ⇒ 48

'1' ⇒ 49

'2' ⇒ 50

'9' ⇒ 57

CHAPTER-3

Conditional Statements

→ first checks Condition, then its statements will work.

Type

(i) if-else (ii) switch

(i) if-else

```
if (condition) {
  // true statements
}
else {
  // false statement
}
```

Program: Note eligibility age>18 allowed.

```
if (age>18) {
  printf("Eligible");
}
else {
  printf("not eligible");
}
```

- Simple if

```
int a=4;
if (5>a){
    printf("a is small");
}
```

- if-else - vote eligibility

- else if

```
if (condition1) {
    // statements.
}
else if (condition 2) {
    //statements
}
```

step by step check ; else (optional)

- "Nested If" → if inside if

Ternary operator (A) For placements.

condition ? True statement : false statement ;

Eg:

```
int age=18;
age >= 18 ? printf("Adult") : printf("not adult");
```


Switch statement :-

```
switch (number) {                    // number or char
case c1 : // do something            case 1 :   | case 'a' :
    break;                           case 2 :   | case 'b' :
case c2 : // do something            case 3 :   | case 'c' :
    break ;
default : // do something
}
```

Program :- Example on switch statement.

(i) input 1-monday 2-Tuesday - - - - 7-sunday

(ii) input m-monday t-tuesday - - - - s-sunday

switch properties

(i) cases can be in any order

(ii) Nested switch (switch inside switch) are allowed.

Comment ctrl + / *

Practice Qs 9

program to check if a student passed or failed
marks > 30 is pass
marks <= 30 is fail

soln: (i) if else
(ii) Ternary operator

Practice Qs 10

write a program to give grades to a student.
marks < 30 is C
30 <= marks < 70 is B
70 <= marks < 90 is A
90 <= marks <= 100 is A+

soln: using else if ladder

Practice Qs 11

Will this code ?

```
int x=2 ;
if (x=1) {
    printf ("x is equal to 1");
} else {
    printf ("x is not equal to 1");
}
```

o/p = ?

My honest Ans :- x is equal to 1 ✓ chetu you are right

but actual Ans is : x is equal to 1

reason x==1 ✗ ⇒ x=1 * assign. and 1 - is non zero number which is True

if (x=0) instead of x=1
then output will be
x is not equal to 1
because 0 will act as False in C programming.

Practice Qs 12

find if a character entered by user is upper case or not

soln: hint

```
if (ch >= 'A' && ch <= 'Z') {
    printf ("uppercase");
} else {
    printf ("not uppercase");
}
```

⇒ we can apply relational operators on character also in 'C' programming because they consists their ASCII Value
A - 65
a - 97

Home work set
✓ ① to check the number is Armstrong ↗ 153 = 1³ + 5³ + 3³ = 1 + 125 + 27 = 153 ✓
✓ ② to check the number is natural

Chapter - 4

Loop Control instructions

→ looping statements means a block of code will be running for 'n' no of times until condition will be satisfied.

→ for, while, do-while loops

→ Break, continue usage.

loops

To repeat some part of the program

Types

(i) For (ii) while (iii) do-while

For loop

```
for (initialization ; condition ; updation) {
    // statements
}

// i - iteration.
```

Ex:
```
for (int i=1 ; i<=5 ; i++) {
    printf ("Hello World\n");
}
```


Practice Qs 13

print the no's from 0 - 10 by for loop

```
for(int i=0 ; i<=10 ; i++){
    printf("%d", i);
}
```

increment operator

pre - increase, then use ++i

post - use, then increase i++

decrement operator

pre - decrease then use --i

post - use, then decrease i--

Special things

→ Increment operator

→ Decrement operator

→ Loop counter / iterator can be float or even character

Eg:
```
for (float i=1.0 ; i<=5.0 ; i++){
    printf ("%f \t", i);
}
```

Sunday 12

```
for (char ch='a' ; ch<='z' ; ch++)
{ printf ("%c \t", ch);
}
// (a-z)
```

2017 February

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28				

2017 March

S	M	T	W	T	F	S
			1	2	3	4
5	6	7	8	9	10	11
12	13	14	15	16	17	18
19	20	21	22	23	24	25
26	27	28	29	30	31	

→ Infinite loop

```
for(int i; ; i++) {
    printf("%d \n", i );
}
```

WHILE Loop

```
while(condition) {
    //something
}
```

Eg:-

```
int i=1;          → declaration
while (i<=5){     → condition
    printf("%d", i);
    i++;          → increment
}
```

Practice Qs 14

(a) print the numbers from 0 to n if n is given by user

```
int i=0, n;
printf("Enter a num: ");
scanf("%d",&n);
while(i<=n){
    printf("%d \n", i);
    i++;
}
```


do-while loop

```
do {
    // do something
} while (condition);
```

Ex:-

```
int i=1;
do {
    printf("%d \n",i);
    i++;
} while(i<=5);
```

Practice Qs 15

(a) print the sum of first n natural numbers.

(b) Also print them in reverse

```
int n=5;
for(int i=n; i>=1; i--) {
    printf("%d\n", i);
}
```

Output: 5 4 3 2 1

Both loops collapsed in one loop

(*)

```
for(int i=1, j=n; i<=n && j>=1; i++, j--) {
    sum=sum+i;          → to find sum
    printf("%d \n", j);  → to print reverse
}
```

NOTE:- // we can declare or initialize two variables in for loop

(*)

```
for(int j=n, j>=1; j--){
    sum=sum+j;
    printf("%d\n", j);
}
```

Practice Qs 16
print the table of a number input by the user

```
int n=5, i;
for(i=1; i<=10; i++){
    printf("%d\n", n*i);
}
```

Break statement :- * used in "switch" statement
↳ exit the loop

Practice Qs 17
Keep taking numbers as input from the user until user enters an odd number.

Hint :- do-while loop

```
int n;
do{
    printf("Enter num:");
    scanf("%d", &n);
}while(n%2==0);
```

```
int n;
do{
    printf("Enter num:");
    scanf("%d", &n);
    if(n%2 != 0){
        break;
    }
}while(1);   // Infinite while loop
```


Practice Qs 18
Keep taking numbers as input from user, until user enters a number is multiple of 7.

```
int n;
do{
    printf("Enter num:");
    scanf("%d", &n);
    if(n%7==0){
        break;
    }
}while(1)   // Infinite while loop
```

Continue Statement

↳ Skip to next iteration

Ex:

```
for (int i=1; i<=5; i++){
    if (i==3){
        continue;
    }
    printf("%d\n", i);
}
```

// 1,2,4,5
// 3 - skip

Practice Qs 19

(*) print all no's from 1 to 10 Except for 6.

```
int i;
for (i=1; i<=10; i++){
    if (i==6){
        continue;
    }
    printf("%d\n", i);
}
```

Practice Qs 20

print all the odd numbers from 5 to 50

```
int i;
for (i=5; i<=50; i++){
    if (i%2!=0){
        printf("%d\n", i);
    }
}
```


Practice Qs 21

print the factorial of a number n

```
int i, fact=1, n;
printf("Enter num:");
scanf("%d", &n);
for (i=1; i<=n; i++){
    fact = fact*i;
}
printf("%d", fact);
```

Practice Qs 22:

print reverse of the table for a number n

```
int n;
printf("Enter num:n");
scanf("%d", &n);
for (int i=10; i>=1; i--){
    printf("%d", n*i);
}
```

Practice Qs 23

① Calculate the sum of all numbers betⁿ 5 & 50 (include 5 & 50)

```
int sum=0;
for (int i=5; i<=50; i++){
    sum = sum+i;
}
printf("sum is %d", sum);
```

Homework set

① ***** / **** / *** / ** / *

print pattern

② check no' is prime or not

③ program to print prime numbers in range

① (L36_filename)

②
```
int n=5; // user input
int count=0, i;
for (i=1; i<=n; i++){
    if (n%i==0){
        count++;
    }
}
if (count==2){
    printf("Prime");
}
else { printf("not prime"); }
```
// logic

Q.no ③

```
#include<stdio.h>
int main(){
    int i,j, count=0;
    int l,h;
    printf("Enter l:");          // l - lower range
    scanf("%d",&l);
    printf("Enter h:");
    scanf("%d",&h);              // h - higher range

    for(i=l; i<=h; i++){
        for(j=1; j<=i; j++){
            if(i%j==0){
                count++;
            }
        }
        if(count==2){
            printf("%d \n",i);
        }
        count=0;
    }
}
```

default

```
if((int)pow(a,n-1)%n==1){
    printf("Prime");
}else{
    printf("not prime");
}
```

a=2, $(a^{n-1})\%n==1$

(ii)
```
int n=5;
for(i=2; i<=sqrt(n); i++){
    if(n%i==0){
        printf("not prime");
        return;
    }
}
printf("prime");
```

prime

(i)
```
int n=5;
for(i=2,i<=n-1;i++){
    if(n%i==0){
        printf("not prime");
        return;
    }
}
printf("prime");
```

(ii)
```
int n=5
for(int i=2; i<=n/2; i++){
    if(n%i==0){
        printf("not prime");
        return;
    }
}
printf("prime");
```