

16 Monday



January 2017

016-349

Week 3

08:00

09:00

10:00

11:00

12:00

01:00

02:00

03:00

04:00

Chores

05:00

January 2017						
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

017-348



Tuesday 17

Week 3

## C. Programming

Basic Advanced

- Vs code / codeblocks / k-develop → code editor (write & edit code) like notebook

→ download vs code for windows

→ download mingw for windows (MINGW) → E → Apply changes → Environment Variable

→ MintGW\bin (path copy) → control panel → system → advanced system settings → Environment Variable → System variables . → path → Edit → New → path copy → OK → OK → OK

→ Our laptop :-

settings → system → About → Advanced system settings → Environment Variable → system variable → path → Edit → New → path paste → OK → OK

→ open vs code → new folder (C-programs) → new file (Hello-world.c)

1st program → SUCCESS.

Hello world. gcc Hello-world.c

gcc -o Hello.o

→ download C/C++ Extension pack

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
29	30	31				

Compiler rules of C are defined

IDE - Integrated development Environment.

18 Wednesday



January 2017  
019-346

1) Programs : Hello-world.C

.C - c program extension

#include <stdio.h>

void main ()

{

    printf ("Hello World");

}

int main (int argc, char \*argv[])

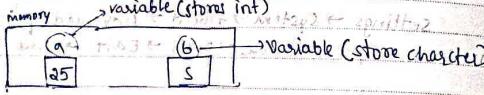
{

    printf ("Hello World");

}

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

- Variables are case-sensitive. `int A=30;` and `int a=40;` are different.
- Alphabet or underscore (start) `int _age=20;` ✓ `int age=18;` ✗
- Special characters are not allowed. (Except <sup>2017</sup>underscores)
- No comma or blank space
- do not start with number

Variables should be meaningful

January 2017  
019-346

January 2017



1970 → old language

Thursday 19

### Data-types

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

Data-type      Format specifier      Size in byte

char              %c              1 byte

int              %d              2 or 4 bytes

float              %f              4 bytes

double              %lf              8 bytes

Eg: `int age = 22;`      signed char ⇒ -128 to 127

`float pi = 3.14;`      unsigned char ⇒ 0 to 255

`char hashtag = '#';`      char also float

`double pi = 3.14;`      double also float

### 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, -2.4, etc.

char constants : 'a', 'b', 'A', '#', etc.

January		February									
S	M	T	W	F	S	S	M	T	W	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	1	2	3	4	5



23 Monday



January 2017

023-342

### Compiling :-

A computer program that translates C code into machine code (Obj) if C code does not have any errors.

hello.c → Compiles → a.exe (Windows)  
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)

$$\text{Area} = \pi \times r \times r \Rightarrow \pi r^2$$

### H.W set

(i) perimeter of rectangle  $\Rightarrow 2(l+b) \Rightarrow 2(l+b)$

(ii) take n from user & output its cube ( $n \times n \times n$ )

(iii) write comments for programs a & b

24.00 ~~writing out for and comparing~~

### Extn

(i) simple interest  $SI = (P \times T \times R) / 100$  #math in

(ii) compound interest  $CI = P \times [1 + R/100]^t - P$  Pow (-,)

(iii)  $F^\circ - C^\circ$   $C^\circ = (F^\circ - 32) \times 5/9$

$$100^\circ C \rightarrow 212^\circ F$$

(iv)  $C^\circ - F^\circ$   $F^\circ = (C^\circ \times 9/5) + 32$

(v) perimeter of circle  $\Rightarrow 2\pi r$

(vi) perimeter of square  $\Rightarrow 4 \times a$

(vii) area of rectangle  $\Rightarrow l \times b$

24.00 ~~writing out for and comparing~~

January 2017



Tuesday 24

024-341

## CHAPTER - 2

### Instructions and Operators

#### Instructions

These are statements in a program.

#### Types

(i) type declaration instruction

(ii) arithmetic instruction

(iii) control instruction.

#### (i) type declaration Instruction

Declare var before using it:

valid

int a = 22;

invalid

int 6 = a + 2; #error

int b = a;

int c = 2;

int d = 5, e; #error

int f;

declaration multiple Variable.

int a,b,c;

R = b = c = 1 ;

int

a=1

b=1

c=1

int a,b,c = 5;

int

a = 5;

b = 5;

c = 5;

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

29 30 31

12 13 14 15 16 17 18 19 20 21 22 23 24 25

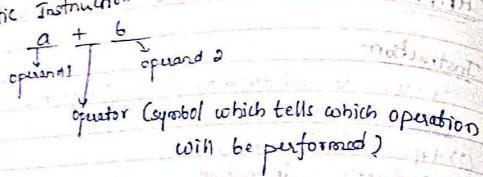
26 27 28

25 Wednesday



January 2017

(i) arithmetic Instruction



NOTE - Single variable on the LHS

valid

$a = b + c;$

$a = b * c;$

$a = b / c;$

invalid

$b + c = a$

$a = b C$

$a = b ^ C$  // No Error

but not  
(Power)

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

$\wedge$  - Bitwise Operator (XOR gate)

Modular Operator

returns remainder of int

$3 \% 2 = 1$

$-3 \% 2 = -1$

$\%$  - not works for float type

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



Thursday 26

Type Conversion.

int op int  $\rightarrow$  int

int op float  $\rightarrow$  float

float op float  $\rightarrow$  float

int op float  $\rightarrow$  float

small container big container  $\rightarrow$  big container (Required)

decimals not required

$a = 3 / 2$

decimals required

$a = 3.0 / 2$

$a = 2 / 3$

$a = 2.0 / 3$

Practice Qs 3 :

solve ?  $\text{int } a = 2.999999;$

P

$P = 2.999999$

Type conversion

$\text{int } a = (\text{inf}) 1.999999;$

$\text{printf} ("y.d", a); \#1$

Implicit  $\rightarrow$  Explicit

Compiler do  $\rightarrow$  User have

for itself  $\rightarrow$  to tell to make conversion

Implicit

float  $b = 1;$  float  $\leftarrow$  int  $\leftarrow$

but

int  $c = 1.2;$  int  $\leftarrow$  float ( $x$ )

so explicit

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
29	30	31				

27 Friday



January 2017

### Operator precedence

08:00 \* / %  
 ↓  
 09:00  $x = 4 + 9 * 10$  // 119  
 ↓  
 10:00 + -  
 ↓  
 11:00  $x = 4 * 3 / 6 * 2$  // 4  
 = (assignment operator)

Associativity (for same precedence)

Left to right

$$x = 4 * 3 / 6 * 2 \Rightarrow 4 //$$

04:00 Practice Qs-4

- (1)  $5 * 2 - 2 + 3$  // 4
- (2)  $5 * 2 / 2 * 3$  // 15
- (3)  $5 * (2 / 2) * 3$  // 15
- (4)  $5 + 2 / 2 * 3$  // 18

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



Saturday 28

### Instructions (Control instructions)

Used to determine flow of program.

- (a) Sequence Control
- (b) Decision Control
- (c) Loop Control
- (d) Case Control.

### Operators

- (a) Arithmetic operators: +, -, \*, /, %
- (b) Relational operators: ==, !=, >, <, >=, <=
- (c) Logical operators: AND, OR, NOT
- (d) Bitwise operators:
- (e) Assignment operators:
- (f) Ternary operators:

### Logical operators

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

Sunday 29

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
29	30	31				

30 Monday		V-GUARD	January 2011
Week 5			010.33
	Procedure	(cont'd from last) 2nd part	
08.00	1	{ +, -, ×, ÷ } arithmetic	
	2	+, - } relational	
09.00	3	<, > } relational	
	4	=, != } relational	
10.00	5		
	6	} logical	
11.00	7		
	⑥	=	
12.00			010.33

**Assignment Operator :-** It changes situation of a variable.

- (i)  $=$ ,  $+ =$ ,  $- =$ ,  $* =$ ,  $/ =$ ,  $\% =$  (Assignment)
- (ii)  $a = b$  (Assignment)
- (iii)  $a = a + 2$  (Assignment)

## Practice Q: 5

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

$\frac{1}{2} \times \frac{1}{2} = \frac{1}{4}$  ~~1/2~~ even odds

(a) even or odd

February 2017

P.Qs-7

- Valid or not

  - i) `int a = 8*8` - Valid (bitwise ~~error~~)
  - ii) `int x ; int y = x ;` - Valid
  - iii) `int x , y=x ;` - invalid (warning)
  - iv) `char str1 = 'x' * 3` - invalid (warning)  
char stores single character

9:50 169 1283 June 19

P.Qs-8 print 1(true) or 0(false) for following statements.

- ④ if it's sunday & its snowing → True  
⑤ if its monday or it's raining → True  
⑥ if a number is greater than 9 & less than 100 → True.

② int issunday = 1;  $\frac{1}{2}$  digit number

int isgnowing = 1;

printf ("y.d", Issur)

Planning, Monitoring & Evaluations, II (and greater)

$$\textcircled{b} \quad \text{inst} \quad \text{isSunday} = 0;$$

bit is raising = 1

```
printf("%d", IsMonday || Israiny); // 1 (operator)
```

2017 March

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

Spring 2001, CDR 44-2001-113

*100-66310-31*

03 Friday



February 17

034-33

HW set

① print the average of 3 numbers.

② to check if given character is digit or not.

③ smallest number

④ isDigit() function

char ch;  
scanf ("%c", &ch);  
printf ("%d", ch>='0' && ch<='9')  
if (ch>='0' && ch<='9')  
 ans ← enterd str + '0' → 48  
else  
 ans ← entered str + '1' → 49  
else  
 ans ← entered str + '2' → 50  
ans = 50 → 57  
ans = 49 → 48  
ans = 48 → 47  
ans = 47 → 46  
ans = 46 → 45  
ans = 45 → 44  
ans = 44 → 43  
ans = 43 → 42  
ans = 42 → 41  
ans = 41 → 40  
ans = 40 → 39  
ans = 39 → 38  
ans = 38 → 37  
ans = 37 → 36  
ans = 36 → 35  
ans = 35 → 34  
ans = 34 → 33  
ans = 33 → 32  
ans = 32 → 31  
ans = 31 → 30  
ans = 30 → 29  
ans = 29 → 28  
ans = 28 → 27  
ans = 27 → 26  
ans = 26 → 25  
ans = 25 → 24  
ans = 24 → 23  
ans = 23 → 22  
ans = 22 → 21  
ans = 21 → 20  
ans = 20 → 19  
ans = 19 → 18  
ans = 18 → 17  
ans = 17 → 16  
ans = 16 → 15  
ans = 15 → 14  
ans = 14 → 13  
ans = 13 → 12  
ans = 12 → 11  
ans = 11 → 10  
ans = 10 → 9  
ans = 9 → 8  
ans = 8 → 7  
ans = 7 → 6  
ans = 6 → 5  
ans = 5 → 4  
ans = 4 → 3  
ans = 3 → 2  
ans = 2 → 1  
ans = 1 → 0

HW due: 2017 S M T W T F S M T W T F  
1 2 3 4 5 6 7 8 9 10 11  
12 13 14 15 16 17 18 19 20 21 22 23  
February

February 2017



Saturday 04

Week-5

CHAPTER - 3

## Conditional Statements

→ first checks Condition - then its statements will work.

10.00 ~~Type~~

(i) if - else

(i) switch - 2212- g/e

(i) if-else

if (Condition) {  
    // true Statement

جی ۶۷۰

Page 5

Q2. What is a false statement (or position) of law?

Progress? Note eligibility age 18 allowed.

```

if (age >= 18) { // condition
    printf ("Eligible");
} else {
    printf ("not eligible");
}

```

• "Habits" (habits) & ("Habits") thing (habits)

<b>2017</b>	<b>March</b>
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	

06 Monday



February 2017

037-328

- Simple if
 

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

```
else if (c>18)
    {
        if (c>18)
            printf("Eligible");
        else (optional)
            // statements
    }
}
```
- else if
 

```
if (condition) {
    statements
    // statements
}
else if (condition 2) {
    statements
}
else (optional)
    // statements
}
```
- Nested If
 

```
if (if inside if, if inside if)
{
    if (if inside if)
        printf("Nested If");
    else
        printf("Nested If");
}
else
    printf("Nested If");
```
- ④ Ternary operator
 

```
condition ? True statement : False statement;
```
- eg
 

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

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				

February 2017



Tuesday 07

Week 6

### Switch Statement :-

```
switch (expression) {
    case c1 : // do something
    case c2 : // do something
    case c3 : // do something
    ...
    default : // do something
}
```

Program :- Examples on switch statement.

(i) input 1-monday / 2-tuesday / 3-wednesday / 4-thursday / 5-friday / 6-saturday / 7-sunday

(ii) input m-monday / t-tuesday / w-wednesday / th-thursday / f-friday / s-saturday / su-sunday

### Switch properties :-

- (i) Cases can be in any order
- (ii) Nested switch (switch inside switch) are allowed.

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	

08 Wednesday

Comment  
Ctrl + /

February 2017

Practice Qs 9  
 program to check if a student passed or failed  
 marks >= 30 is pass  
 marks <= 30 is fail  
 Solution:  
 (i) if else ladder  
 (ii) Ternary operator

Practice Qs 10  
 write a program to give grades to a student.  
 marks < 80 is C  
 80 <= marks < 70 is B  
 70 <= marks <= 90 is A  
 90 <= marks <= 100 is A+  
 Solution: Using else if ladder

Practice Qs 11

Will this code? What you get and why?

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

O/P = ?  
 My honest Ans :- x is equal to 1 ✓ chetu you are right  
 but actual Ans is : x is equal to 1  
 reason  $x == 1$   $\rightarrow x = 1$   
 \*Assign. and 1 - is non zero number  
 which is True

February						
S	M	T	W	F	S	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				

February 2017

If ( $x=0$ ) instead of  $x=1$   
 then output will be  
 $x \neq 0$   
 because 0 will act as False in C programming.

Practice Qs 12  
 find if a character entered by user is upper case or not  
 seen: hint  
 If ( $ch \geq 'A'$  &  $ch \leq 'Z'$ ) {  
 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  
 Armstrong number  
 Armstrong number =  $1^3 + 5^3 + 3^3 = 153$

Work set  
 ① to check the number is Armstrong  
 ② to check the number is natural

March						
S	M	T	W	F	S	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				

10 Friday



February 2017

## Chapter - 4

### loop Control instructions

- Loop statements mean 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");
```

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

February 2017



Saturday 11

Week 6

#### Practice Qs 33

print the no's from 10-20 by for loop  
for (int i=0; i<=10; i++)  
 printf("%d", i);

#### increment operator

pre - increase, then use  
post - use, then increase

#### decrement operator

pre - decrease then use  
post - use, then decrease

02:00

Special things

#### Increment operator

#### Decrement operator

→ Loop counter / iterator can be float or even character

Ex:

```
for (float i=1.0; i<=5.0; i++) {
```

```
    printf("%f", i);
```

Sunday 12

```
for (char ch='a'; ch<='z'; ch++) {
```

```
    printf("%c", ch);
```

// Ca-Z

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				

13 Monday



February 2017

→ Infinite loop

```
08:00 for (int i=5; i>0; i++) {
    printf("%d\n", i);
}
```

WHILE Loop

```
11:00 while (condition) {           Eg:   int i=1;
    //something                         while (i<=5) {
}                                         printf("%d", i);
                                         i++;
                                         }           → increment
```

Practice Qs 14

⑩ print the numbers from 0 to n if n is given by user

```
04:00 int i=0, n;
05:00 printf("Enter a number:");
06:00 scanf("%d", &n);
07:00 while (i<=n) {
08:00     printf("%d\n", i);
09:00     i++;
}
```

February						
S	M	T	W	F	S	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

February 2017



Tuesday 14

do - while loop

```
08:00 do {                                dt is a session
    int i=1;                                + i loops
    cout << i;                                do { i = 1
}                                do something
                                         & while (condition);      printf("%d\n", i);
                                         i++;                      } while (i<=5);
```

Practice Qs 15

⑪ print the sum of first n natural numbers.

⑫ also print them in reverse

```
01:00 int n=5;
02:00 for (int i=n; i>=1; i--) {
03:00     printf("%d\n", i);
04:00 }
05:00
06:00
07:00
08:00
09:00
10:00
11:00
12:00
13:00
14:00
15:00
16:00
17:00
18:00
19:00
20:00
21:00
22:00
23:00
24:00
25:00
26:00
27:00
28:00
29:00
30:00
31:00
32:00
33:00
34:00
35:00
36:00
37:00
38:00
39:00
40:00
41:00
42:00
43:00
44:00
45:00
46:00
47:00
48:00
49:00
50:00
51:00
52:00
53:00
54:00
55:00
56:00
57:00
58:00
59:00
60:00
61:00
62:00
63:00
64:00
65:00
66:00
67:00
68:00
69:00
70:00
71:00
72:00
73:00
74:00
75:00
76:00
77:00
78:00
79:00
80:00
81:00
82:00
83:00
84:00
85:00
86:00
87:00
88:00
89:00
90:00
91:00
92:00
93:00
94:00
95:00
96:00
97:00
98:00
99:00
100:00
101:00
102:00
103:00
104:00
105:00
106:00
107:00
108:00
109:00
110:00
111:00
112:00
113:00
114:00
115:00
116:00
117:00
118:00
119:00
120:00
121:00
122:00
123:00
124:00
125:00
126:00
127:00
128:00
129:00
130:00
131:00
132:00
133:00
134:00
135:00
136:00
137:00
138:00
139:00
140:00
141:00
142:00
143:00
144:00
145:00
146:00
147:00
148:00
149:00
150:00
151:00
152:00
153:00
154:00
155:00
156:00
157:00
158:00
159:00
160:00
161:00
162:00
163:00
164:00
165:00
166:00
167:00
168:00
169:00
170:00
171:00
172:00
173:00
174:00
175:00
176:00
177:00
178:00
179:00
180:00
181:00
182:00
183:00
184:00
185:00
186:00
187:00
188:00
189:00
190:00
191:00
192:00
193:00
194:00
195:00
196:00
197:00
198:00
199:00
200:00
201:00
202:00
203:00
204:00
205:00
206:00
207:00
208:00
209:00
210:00
211:00
212:00
213:00
214:00
215:00
216:00
217:00
218:00
219:00
220:00
221:00
222:00
223:00
224:00
225:00
226:00
227:00
228:00
229:00
230:00
231:00
232:00
233:00
234:00
235:00
236:00
237:00
238:00
239:00
240:00
241:00
242:00
243:00
244:00
245:00
246:00
247:00
248:00
249:00
250:00
251:00
252:00
253:00
254:00
255:00
256:00
257:00
258:00
259:00
260:00
261:00
262:00
263:00
264:00
265:00
266:00
267:00
268:00
269:00
270:00
271:00
272:00
273:00
274:00
275:00
276:00
277:00
278:00
279:00
280:00
281:00
282:00
283:00
284:00
285:00
286:00
287:00
288:00
289:00
290:00
291:00
292:00
293:00
294:00
295:00
296:00
297:00
298:00
299:00
300:00
301:00
302:00
303:00
304:00
305:00
306:00
307:00
308:00
309:00
310:00
311:00
312:00
313:00
314:00
315:00
316:00
317:00
318:00
319:00
320:00
321:00
322:00
323:00
324:00
325:00
326:00
327:00
328:00
329:00
330:00
331:00
332:00
333:00
334:00
335:00
336:00
337:00
338:00
339:00
340:00
341:00
342:00
343:00
344:00
345:00
346:00
347:00
348:00
349:00
350:00
351:00
352:00
353:00
354:00
355:00
356:00
357:00
358:00
359:00
360:00
361:00
362:00
363:00
364:00
365:00
366:00
367:00
368:00
369:00
370:00
371:00
372:00
373:00
374:00
375:00
376:00
377:00
378:00
379:00
380:00
381:00
382:00
383:00
384:00
385:00
386:00
387:00
388:00
389:00
390:00
391:00
392:00
393:00
394:00
395:00
396:00
397:00
398:00
399:00
400:00
401:00
402:00
403:00
404:00
405:00
406:00
407:00
408:00
409:00
410:00
411:00
412:00
413:00
414:00
415:00
416:00
417:00
418:00
419:00
420:00
421:00
422:00
423:00
424:00
425:00
426:00
427:00
428:00
429:00
430:00
431:00
432:00
433:00
434:00
435:00
436:00
437:00
438:00
439:00
440:00
441:00
442:00
443:00
444:00
445:00
446:00
447:00
448:00
449:00
450:00
451:00
452:00
453:00
454:00
455:00
456:00
457:00
458:00
459:00
460:00
461:00
462:00
463:00
464:00
465:00
466:00
467:00
468:00
469:00
470:00
471:00
472:00
473:00
474:00
475:00
476:00
477:00
478:00
479:00
480:00
481:00
482:00
483:00
484:00
485:00
486:00
487:00
488:00
489:00
490:00
491:00
492:00
493:00
494:00
495:00
496:00
497:00
498:00
499:00
500:00
501:00
502:00
503:00
504:00
505:00
506:00
507:00
508:00
509:00
510:00
511:00
512:00
513:00
514:00
515:00
516:00
517:00
518:00
519:00
520:00
521:00
522:00
523:00
524:00
525:00
526:00
527:00
528:00
529:00
530:00
531:00
532:00
533:00
534:00
535:00
536:00
537:00
538:00
539:00
540:00
541:00
542:00
543:00
544:00
545:00
546:00
547:00
548:00
549:00
550:00
551:00
552:00
553:00
554:00
555:00
556:00
557:00
558:00
559:00
560:00
561:00
562:00
563:00
564:00
565:00
566:00
567:00
568:00
569:00
570:00
571:00
572:00
573:00
574:00
575:00
576:00
577:00
578:00
579:00
580:00
581:00
582:00
583:00
584:00
585:00
586:00
587:00
588:00
589:00
590:00
591:00
592:00
593:00
594:00
595:00
596:00
597:00
598:00
599:00
600:00
601:00
602:00
603:00
604:00
605:00
606:00
607:00
608:00
609:00
610:00
611:00
612:00
613:00
614:00
615:00
616:00
617:00
618:00
619:00
620:00
621:00
622:00
623:00
624:00
625:00
626:00
627:00
628:00
629:00
630:00
631:00
632:00
633:00
634:00
635:00
636:00
637:00
638:00
639:00
640:00
641:00
642:00
643:00
644:00
645:00
646:00
647:00
648:00
649:00
650:00
651:00
652:00
653:00
654:00
655:00
656:00
657:00
658:00
659:00
660:00
661:00
662:00
663:00
664:00
665:00
666:00
667:00
668:00
669:00
670:00
671:00
672:00
673:00
674:00
675:00
676:00
677:00
678:00
679:00
680:00
681:00
682:00
683:00
684:00
685:00
686:00
687:00
688:00
689:00
690:00
691:00
692:00
693:00
694:00
695:00
696:00
697:00
698:00
699:00
700:00
701:00
702:00
703:00
704:00
705:00
706:00
707:00
708:00
709:00
710:00
711:00
712:00
713:00
714:00
715:00
716:00
717:00
718:00
719:00
720:00
721:00
722:00
723:00
724:00
725:00
726:00
727:00
728:00
729:00
730:00
731:00
732:00
733:00
734:00
735:00
736:00
737:00
738:00
739:00
740:00
741:00
742:00
743:00
744:00
745:00
746:00
747:00
748:00
749:00
750:00
751:00
752:00
753:00
754:00
755:00
756:00
757:00
758:00
759:00
760:00
761:00
762:00
763:00
764:00
765:00
766:00
767:00
768:00
769:00
770:00
771:00
772:00
773:00
774:00
775:00
776:00
777:00
778:00
779:00
780:00
781:00
782:00
783:00
784:00
785:00
786:00
787:00
788:00
789:00
790:00
791:00
792:00
793:00
794:00
795:00
796:00
797:00
798:00
799:00
800:00
801:00
802:00
803:00
804:00
805:00
806:00
807:00
808:00
809:00
810:00
811:00
812:00
813:00
814:00
815:00
816:00
817:00
818:00
819:00
820:00
821:00
822:00
823:00
824:00
825:00
826:00
827:00
828:00
829:00
830:00
831:00
832:00
833:00
834:00
835:00
836:00
837:00
838:00
839:00
840:00
841:00
842:00
843:00
844:00
845:00
846:00
847:00
848:00
849:00
850:00
851:00
852:00
853:00
854:00
855:00
856:00
857:00
858:00
859:00
860:00
861:00
862:00
863:00
864:00
865:00
866:00
867:00
868:00
869:00
870:00
871:00
872:00
873:00
874:00
875:00
876:00
877:00
878:00
879:00
880:00
881:00
882:00
883:00
884:00
885:00
886:00
887:00
888:00
889:00
890:00
891:00
892:00
893:00
894:00
895:00
896:00
897:00
898:00
899:00
900:00
901:00
902:00
903:00
904:00
905:00
906:00
907:00
908:00
909:00
910:00
911:00
912:00
913:00
914:00
915:00
916:00
917:00
918:00
919:00
920:00
921:00
922:00
923:00
924:00
925:00
926:00
927:00
928:00
929:00
930:00
931:00
932:00
933:00
934:00
935:00
936:00
937:00
938:00
939:00
940:00
941:00
942:00
943:00
944:00
945:00
946:00
947:00
948:00
949:00
950:00
951:00
952:00
953:00
954:00
955:00
956:00
957:00
958:00
959:00
960:00
961:00
962:00
963:00
964:00
965:00
966:00
967:00
968:00
969:00
970:00
971:00
972:00
973:00
974:00
975:00
976:00
977:00
978:00
979:00
980:00
981:00
982:00
983:00
984:00
985:00
986:00
987:00
988:00
989:00
990:00
991:00
992:00
993:00
994:00
995:00
996:00
997:00
998:00
999:00
1000:00
1001:00
1002:00
1003:00
1004:00
1005:00
1006:00
1007:00
1008:00
1009:00
1010:00
1011:00
1012:00
1013:00
1014:00
1015:00
1016:00
1017:00
1018:00
1019:00
1020:00
1021:00
1022:00
1023:00
1024:00
1025:00
1026:00
1027:00
1028:00
1029:00
1030:00
1031:00
1032:00
1033:00
1034:00
1035:00
1036:00
1037:00
1038:00
1039:00
1040:00
1041:00
1042:00
1043:00
1044:00
1045:00
1046:00
1047:00
1048:00
1049:00
1050:00
1051:00
1052:00
1053:00
1054:00
1055:00
1056:00
1057:00
1058:00
1059:00
1060:00
1061:00
1062:00
1063:00
1064:00
1065:00
1066:00
1067:00
1068:00
1069:00
1070:00
1071:00
1072:00
1073:00
1074:00
1075:00
1076:00
1077:00
1078:00
1079:00
1080:00
1081:00
1082:00
1083:00
1084:00
1085:00
1086:00
1087:00
1088:00
1089:00
1090:00
1091:00
1092:00
1093:00
1094:00
1095:00
1096:00
1097:00
1098:00
1099:00
1100:00
1101:00
1102:00
1103:00
1104:00
1105:00
1106:00
1107:00
1108:00
1109:00
1110:00
1111:00
1112:00
1113:00
1114:00
1115:00
1116:00
1117:00
1118:00
1119:00
1120:00
1121:00
1122:00
1123:00
1124:00
1125:00
1126:00
1127:00
1128:00
1129:00
1130:00
1131:00
1132:00
1133:00
1134:00
1135:00
1136:00
1137:00
1138:00
1139:00
1140:00
1141:00
1142:00
1143:00
1144:00
1145:00
1146:00
1147:00
1148:00
1149:00
1150:00
1151:00
1152:00
1153:00
1154:00
1155:00
1156:00
1157:00
1158:00
1159:00
1160:00
1161:00
1162:00
1163:00
1164:00
1165:00
1166:00
1167:00
1168:00
1169:00
1170:00
1171:00
1172:00
1173:00
1174:00
1175:00
1176:00
1177:00
1178:00
1179:00
1180:00
1181:00
1182:00
1183:00
1184:00
1185:00
1186:00
1187:00
1188:00
1189:00
1190:00
1191:00
1192:00
1193:00
1194:00
1195:00
1196:00
1197:00
1198:00
1199:00
1200:00
1201:00
1202:00
1203:00
1204:00
1205:00
1206:00
1207:00
1208:00
1209:00
1210:00
1211:00
1212:00
1213:00
1214:00
1215:00
1216:00
1217:00
1218:00
1219:00
1220:00
1221:00
1222:00
1223:00
1224:00
1225:00
1226:00
1227:00
1228:00
1229:00
1230:00
1231:00
1232:00
1233:00
1234:00
1235:00
1236:00
1237:00
1238:00
1239:00
1240:00
1241:00
1242:00
1243:00
1244:00
1245:00
1246:00
1247:00
1248:00
1249:00
1250:00
1251:00
1252:00
1253:00
1254:00
1255:00
1256:00
1257:00
1258:00
1259:00
1260:00
1261:00
1262:00
1263:00
1264:00
1265:00
1266:00
1267:00
1268:00
1269:00
1270:00
1271:00
1272:00
1273:00
1274:00
1275:00
1276:00
1277:00
1278:00
1279:00
1280:00
1281:00
1282:00
1283:00
1284:00
1285:00
1286:00
1287:00
1288:00
1289:00
1290:00
1291:00
1292:00
1293:00
1294:00
1295:00
1296:00
1297:00
1298:00
1299:00
1300:00
1301:00
1302:00
1303:00
1304:00
1305:00
1306:00
1307:00
1308:00
1309:00
1310:00
1311:00
1312:00
1313:00
1314:00
1315:00
1316:00
1317:00
1318:00
1319:00
1320:00
1321:00
1322:00
1323:00
1324:00
1325:00
1326:00
1327:00
1328:00
1329:00
1330:00
1331:00
1332:00
1333:00
1334:00
1335:00
1336:00
1337:00
1338:00
1339:00
1340:00
1
```

15 Wednesday



February 2017

Practice Qs 16

08:00 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);  
}

12:00 Break statement: - used in "switch" statement

01:00 ↴ exit-the loop

Practice Qs 17

03:00 Keep taking numbers as input from the user until user enters an odd number

04:00 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);  
} while (n%2 != 0);  
  
if (n%2 == 0)  
    break;
```

4  
ywhile (1); //Infinite loop

February  
2017

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											

February 2017



Thursday 16

Week-7

04:00 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 loop
```

2017 March  
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

17 Friday



February 2017

Week-7 Continue Statement

→ Skip to next iteration using a break

```
09:00    for (int i=1; i<=5; i++) {  
09:00        if (i==3) {  
09:00            continue; // skip  
09:00        } // 1, 2, 4, 5  
09:00        printf("%d\n", i);  
09:00    } // 2, 3, 4, 5
```

Practice Qs 19

④ print all no's from 1 to 10 Except for 6.

```
03:00    int i;  
03:00    for (i=1; i<=10; i++) {  
03:00        if (i==6) {  
03:00            continue;  
03:00        }  
03:00        printf("%d\n", i);  
03:00    } // 1, 2, 3, 4, 5, 7, 8, 9, 10
```

Practice Qs 20

print all the odd numbers from 5 to 50

```
04:00    int i;  
04:00    for (i=5; i<=50; i++) {  
04:00        if (i%2!=0) {  
04:00            printf("%d\n", i);  
04:00        } // 5, 7, 9, 11, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31
```

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

February 2017



Saturday 18

Week-7

04:316 Practice Qs 21

```
08:00 print the factorial of a number n  
08:00    int i, fact=1, n;  
08:00    printf("Enter num:");  
08:00    scanf("%d", &n);  
09:00    for (i=1; i<=n; i++) {  
09:00        fact = fact*i;  
10:00    }  
10:00    printf("%d", fact);  
10:00
```

01:00 Practice Qs 22:

```
02:00 print reverse of the table for a number n  
02:00    int n;  
02:00    printf("Enter num:");  
02:00    scanf("%d", &n);  
03:00    for (int i=10; i>=1; i--) {  
03:00        printf("%d", n-i);  
03:00    } // 10, 9, 8, 7, 6, 5, 4, 3, 2, 1
```

Sunday 19

February							March						
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				

22 Wednesday



February 2017

Practice Qs 23

① calculate the sum of all numbers between 5 & 50 (include 5 & 50)

```

08:00 int sum=0;
09:00 for (int i=5; i<=50; i++) {
10:00     sum = sum + i;
11:00 }
12:00 printf ("sum is %d", sum);
    
```

Homework set

① ~~\* \* \* \*~~ ② check no' is prime or not  
~~\* \* \* \*~~  
~~\* \* \* \*~~ ③ program to print prime numbers in range  
~~\* \* \* \*~~  
 print pattern

④ ✓ (3c-filename)

⑤ 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");
    }
    
```

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

2017 March  
 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

February 2017



Thursday 23

Week 8

① #include <stdio.h>

```

08:00 int main() {
09:00     int i, j, count = 0;
10:00     printf ("Enter l : ");
11:00     scanf ("%d", &l); // l - lower range
12:00     printf ("Enter h : ");
13:00     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);
 } else {
 printf ("prime");
 }

int n=5;

```

08:00 for (int i=2; i<n; i++) {
09:00     if (n % i == 0) {
10:00         printf ("not prime");
11:00         return;
12:00     }
13:00     printf ("prime");
    
```