C-Programming and DSA

2-3 month

Topics

Programming
- Basics(Operators, Loops, Macros, Switch etc.)
- Storage Classes and Scopes
- Functions
- Pointer and Strings
- Arrays
- Recursion
- Structure, Union and Dynamic memory Allocation

Data Structure
- Linked List
- Stacks
- Queues
- Tree(BST, AVL)
- Hashing

"C-Test your Aptitude" By Venugopal & Chandrakanta
"C-Langage" by Denis Ritchie
"GATE previous years Question"

Storageclass → Static, Function, Array, Pointer, News → Books'

DSA

Mr. Tanenbaum

Data Structure → Geeks for Geeks (GFG)

Datastructure & Algo. By "Narshima Karumanchi"

Thomas H. Coremen

GATE Weightage

Marks 12 to 14

GATE - 15 Qus

99.24

C-Programming and DSA

Today Class Topics

- Basics of C-programming
- Operators
- Loops
- if else
- Switch
- Macros

C-Programming and DSA

Q-1:

```
main()
{ int x=5, y=10;
  if(x!=5)
      if(y==10)
          Pf("Hi");
  else
      Pf("Hello");
}
```

O/p = ?

Double ≥ float

Q-2:

```
main()
{ float a=0.5, b=0.7;
  if(b<0.7)
  { if(a<0.5)
        Pf("Hi");
    else
        Pf("Hello");
  }
  else
      Pf("Bye");
}
```

O/p = ? Hello

C-Programming and DSA

Operator Precedance Table:

Description	Operator	Associativity
Function expression Array Expression Structure Operator	( ) [ ] → .	L to R
Unary minus Increment/Decrement One's Compliment Negation Address of value of Address Type Cast Size in Byte	— ++/-- ~ ! & * (type) sizeof	R to L
Multiplication Division Modulus	* / %	L to R
Addition Substraction	+ —	L to R
Left shif Right shif	<< >>	L to R
Less than, Less than equal to Greater than, Greater than equal to	<, <= >, >=	L to R
Equal to Not Equal to	== !=	L to R
Bitwise AND	&	L to R
Bitwise exclusive OR	^	L to R
Bitwise OR	|	L to R
Logical AND	&&	L to R
Logical OR	||	L to R
Conditional	? :	R to L
Assignment	=, *=, /=, %=, += -=, &=, ^=, <<=, >>=	R to L
Comma	,	R to L

Higher

lower

Ex:- 9*3/2%5

27/2%5

13%5 = 3

L to R

9-5+4 => 9-9 = 0

4+4 = 8

C-Programming and DSA

Expression	Result (with respect to C- Language Compiler)
5/2	2
5.0/2	2.5
5.0/2.0	2.5
2/5	0 ✓
2.0/5	0.4
2.0/5.0	0.4

2/5 = 0.4

NOTE:- if both are integer then output will be integer, if any one is float then output will be float

C-Programming and DSA

(<, <=, >, >=, !=) &&, ||

Relational & logical operators:

while( )

TRUE	FALSE
1	0
10	0.0
-10	'\0' → ASCII → 0
6.5	NULL
-3.5	
0.6	

0.6, -4, 10.4, 2, 1 } TRUE

0 } false

TRUE

a = 5>4 = 1

pf(a) = 1

a = 5>4 + 3<4
    1   + 1

a = ? a = 2

a = 1>3 , a = ?
  false   a = 0

C-Programming and DSA

Expression		Assign to int	Assign to float
5	5	5	5.0
5/2	2	2	2.0
5.0/2	2.5	2	2.5
5.0/2.0	2.5	2	2.5
2/5	0	0	0.0
2.0/5	0.4	0	0.4
2.0/5.0	0.4	0	0.4

C-Programming and DSA

Modulus operator (%) :-

1) $15 \% 7 = ? \Rightarrow 1$

2) $-15 \% +7 = ? \Rightarrow -1$

3) $+15 \% -7 = ? \Rightarrow +1$

4) $-15 \% -7 = ? \Rightarrow -1$

5) $-15.5 \% +7 = ? \Rightarrow$ Error

6) $+15.5 \% -7 = ? \Rightarrow$ Error

7) $-7 \% 15 = ? \Rightarrow -7$

8) $+7 \% -15 = ? \Rightarrow +7$

```
 7) 15 (2
    14
    ---
     1
```

```
+7) -15 (-2
    -14
    ---
     -1
```

```
-7) 15 (-2
    -14
    ---
     +1
```

```
-7) -15 (+2
    -14
    ---
     -1
```

C-Programming and DSA

NOTE:

1) Modulus always gives numerator sign ✓
2) Modulus doesn't work on float values, it works only on the integers
3) if the value is small without sign, then it gives same value as output.

C-Programming and DSA

Format specifiers:-

- %d - int
- %f - float
- %c - char
- %s - String
- %u - address

Ex:- `a = 2*3/4 + 2.0/5 + 8/5;` ⇒ 2.4

(6/4 + 2.0/5 + 8/5)

0.4

`printf("%d", a);` ⇒ 2

o/p = ?

a = 8

Ex:-

```
Void main()
{ int a=5;
  if(a==8)
   {printf("Hi");}
  else
   {printf("Hello");}
  printf("%d",a);
}
```

o/p = ?

Hi

if(a==8)

a = 8

C-Programming and DSA

Printf():-

```
Void main()
{ int x;
  x = printf("Hello");
  printf("%d", x);
}
```

O/p = ?

Hello5

C-Programming and DSA

NOTE:-

1) Printf() always returns integer i.e No. of Symbols displayed on the screen

Ex:-
```
void main()
{
  int x;
  x = (printf("Hello"), printf("friends"));
  Pf("%d", x);
}
```

Hellofriends7

O/P = ?

```
x = (2, 3, 4, 1);
```

x = 1

```
x = 2, 3, 4;
```

x = 2

C-Programming and DSA

Scanf(): Scanf() always return integers, i.e no. of proper inputs given, according to format specifier

Ex:
```
main()
{ int i, J, K, m;
  m = scanf("%d%d%d", &i, &J, &K);
  pf("%d", m);
}
```

3

Input	P
*, $	0
#, 0	1
5.5, $	0
10, 20	2

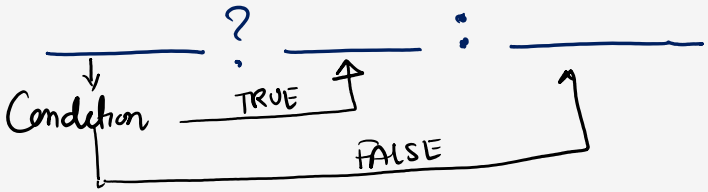
o/p = ? (3)

Ex:
```
void main()
{ int x, y, P;
  printf("Enter two proper integers");
  P = scanf("%d%d", &x, &y);
  if(P==2)
  { printf("proper input gives i.e %d%d", x, y);
  }
  else
  { printf("wrong input gives");
  }
}
```

10, 20

C-Programming and DSA

Ternary operator:-



? :

NOTE:

1) In the ternary operator there should be equal no. of colon's and question marks, otherwise It gives error

2) Every colon should match with the just before question mark

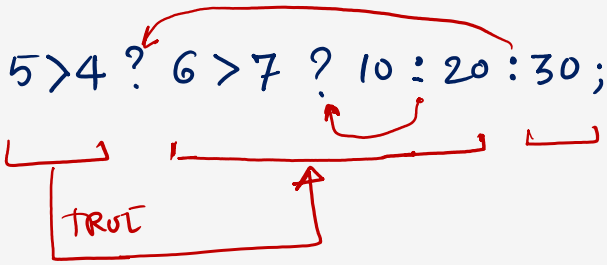
3) Every question mark followed by colon not immediately but following

Ex:- ⇒ a = 3 > 4 ? 10 : 8 > 7 ? 20 : 30;

a = 20 Ans

C-Programming and DSA

Ex1 `a = 5>4 ? 6>7 ? 10 : 20 : 30;` = ?



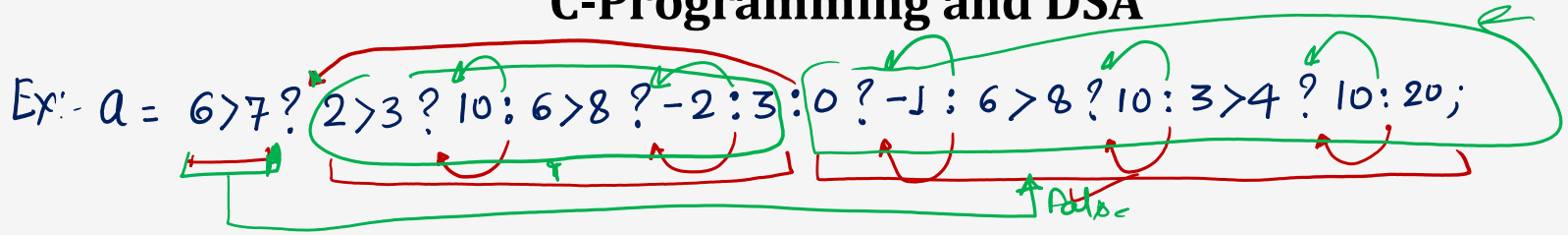
`a = 6>7 ? 10 : 20` = 20 ⇒ a = 20

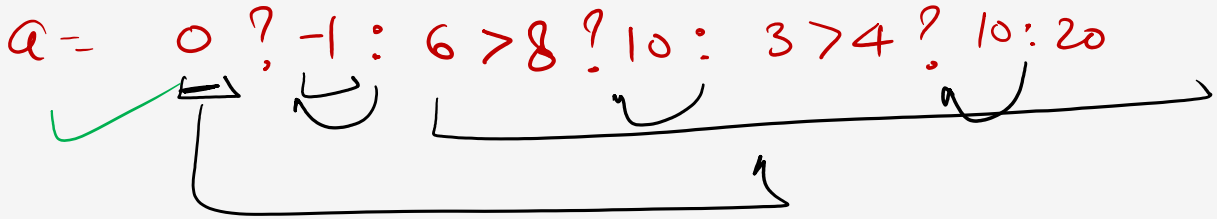
Ex3 `a = 2>3 ? 5>4 ? 10 : 30 : 8>7 ? 30 : 40;` = ?

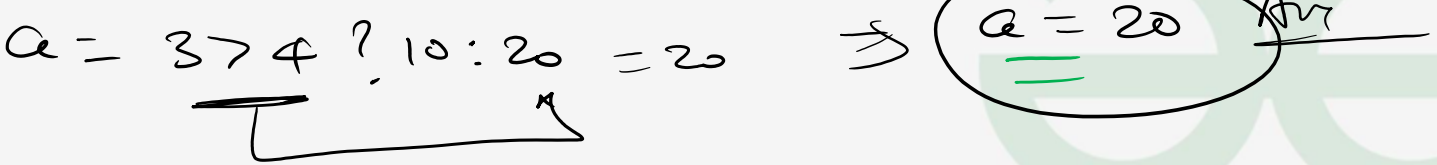
`a = 8>7 ? 30 : 40`

a = 30

C-Programming and DSA







C-Programming and DSA

Question:- WAP to find maximum of 3 no's using ternary operator

a max, a, b, c;

max = a > b ? a > c ? a : c : b > c ? b : c

TRUE

a = 2
b = 4
c = 3

C-Programming and DSA

GATE:

Which Combination of the integers variables x, y & z makes variable 'a' get the value 4 in the following expression?

```
a= x>y ? x>z ? x : z : y>z ? y : z;
```

a) x=6, y=5, z=3

b) x=6, y=3, z=5

c) x=5, y=4, z=5

d) x=3, y=4, z=2

C-Programming and DSA

Pre/Post Increment/Decrement:-

++a => a = a+1

a | 10

b |

PRE	Post	PRE	Post
a=10; 11 b=++a;	a=10; b=a++;	a=10; 9 b=--a;	a=10; b=a--;
a=11 b=11	a=11 b=10	a=9 b=9	a=9 b=10

C-Programming and DSA

a=10; //
 a++;
 pf(a);
	a=10; //
	 ++a;
	 pf(a);
		a=10; //
		 pf(a++);
			a=10; //
			 pf(++a);

o/p = 11	o/p = 11	o/p = 10	o/p = 11

C-Programming and DSA

GATE:-

o/p = ?

```
main()
{ int m = 10;
  int n, n1;
  n = ++m;
  n1 = m++;
  n--;
  --n1;
  n -= n1; // n = n - n1 = 0
  Pf("%d", n); // 0
}
```

m 10 11 12

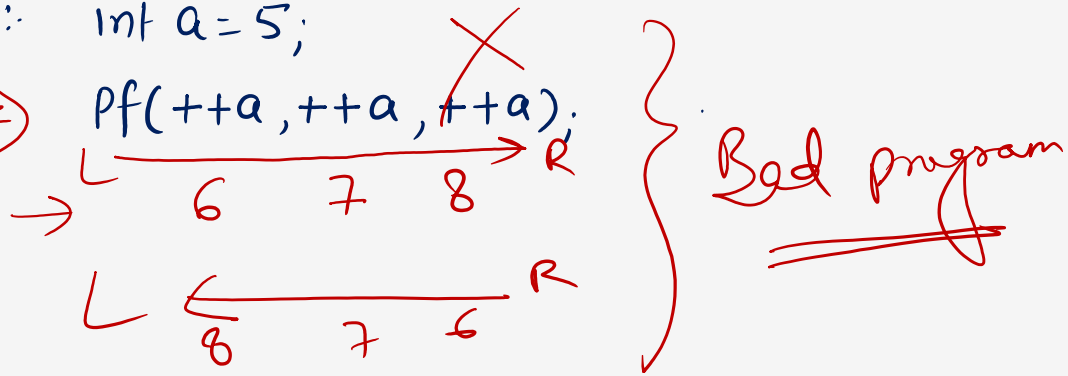
n 11 10

n1 11 10

C-Programming and DSA

Question: int a = 5;

⇒ pf(++a, ++a, ++a);



Bad program

Ex:- int a=5;

pf(a+1, a+2, a+3);

→ 6 7 8

6 7 8

C-Programming and DSA

Question:

```
main()
{
  int a=5;
  pf("%d", ++(a++));
  pf("%d", (a+1)++);
}
```

++5

5

7 + 1

5 = 5 + 1

x = 5 + 1

a 6

++6

5 = 5 + 1

l-value error

C-Programming and DSA

```
a=5;
f(a++);
P.v = 5
a.v = 6
```

```
a=5;
f(++a);
P.v = 6
a.v = 6
```

```
a=5;
f(a+1);
P.v = 6
a.v = 5
```

a++

C-Programming and DSA

Short Circuit Evaluation:

Question: a = 1, b = 1, c = 1, d = 1

```
a = (b++ > 1 || c++ > 1) && d++ > 1;
```

1 > 1 || 1 > 1

FALSE && ⇒ False

b = 2, c = 2

a = 0

What is the value of a, b, c, d ?

0 2 2 1

C-Programming and DSA

Question: int i = 0, j = -1, k = -1, l = 2, m;

m = ((i++ && j++ && k++) || (l++)), = 1

Pf (i, j, k, l, m);

Input	output (i, j, k, l, m)
orignal	(0, 0, 0, 2, 1)
if k=0	(0 0 1 3 1)
if e =0	(1, -1, -1, 3, 1)

C-Programming and DSA

Question:-

```
main()
{
  int i=1, j=2, k=3, m;
  m = i++ && ++j || ++k;
  Pf("%d%d%d%d", i, j, k, m);
}
```

O/P = ?

C-Programming and DSA

Question:-

```
main()
{ int i=2, j=3, k=4, m;
  m = ++i || (++j && ++k);
  Pf(i, j, k, m);
```

O/p = ?

C-Programming and DSA

Bitwise operators :-

Operator	meaning
✓ &	Bitwise AND
✓ |	Bitwise OR
✓ ^	Bitwise XOR
✓ <<	Left shift
✓ >>	Right shift
~	

⇒ 01001011

⇒ & 00010101 = 00000001 ⇒ (1)

01001011 | 00010101 = 01011111

01001011 ^ 00010101 = 01011110

Ex:-
```
main()
{
  int a=25,b=12;
  printf("%d",a&b);  // 8
  printf("%d",a|b);  // 29
}
```

a = 25 = 00011001

b = 12 = 00001100

a&b = 00001000

a|b = 00011101 (1+4+8+16)

C-Programming and DSA

Bitwise Compliment:-(~)

~0100111

1011000

0|0

110

Ex:

```
int i = 2;
i = ~i;
Pf("%d", i);
```

⇒

0010

1101

)101

−(011)

−3 Ans

Ex:

```
int i = 5;
i = ~i;
Pf("%d", i);
```

0101

)010

−(110)

⇒ −6

C-Programming and DSA

<<

⇒ Left Shift (<<):- Left Shift operator shift all the bits towards Left, by certain No. of specified bits,

Every Left Shift Perform multiply by 2

Ex: if x = 2

2 = 0010 <<1

4 = 0100 <<1

8 = 1000

if x = 5

5 = 0101 <<1

10 = 1010 <<1

20 = 10100

$x = x << K$ times

$x = x * 2^K$

Ex: int x = 2;

x = x<<3;

Pf(x); // $x = x * 2^3$

$x = 2 * 2^3 = 16$

0010

0100

C-Programming and DSA

Right Shift (>>) :- Right shift operator shifts all the bits towards right by a certain number of specified bits, Every Right shift Perform divide by 2

$x = x >> k$

$x = \frac{x}{2^k}$

Ex:- Let x = 20

20 : 0 1 0 1 0 0 ≥ 1

/2 → 10 : 0 0 1 0 1 0 ≥ 1

/2 → 5 : 0 0 0 1 0 1 ≥ 1

2 : 0 0 0 0 1 0

```
int x = 20;
x = x >> 3;
pf(x);
```

⇒ 2

$x = \frac{20}{2^3} = \frac{2}{8} \Rightarrow 2$

C-Programming and DSA

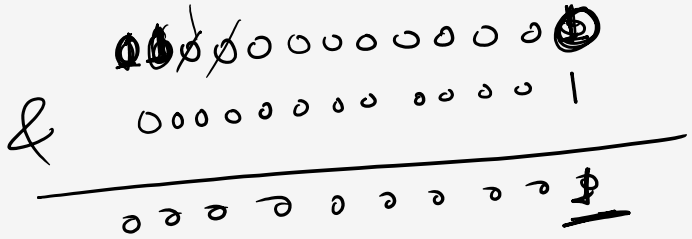
GATE

Ex:- Void main()
{ int x = 4096;
While (x)
{ if (x & 1)
printf("GATECSE2025");
x >>= 1;
}
}

$x >>= 1$,

$x = x >> 1$

2^{12}



GATECSE 2025

How many times String "GATECSE2025" will print (1) Any ?

C-Programming and DSA

Datatype & operator:-

Char : 1B ⇒ 8 bit

-2^{n-1} to $2^{n-1}-1$

-2^{7} to $2^{7}-1$

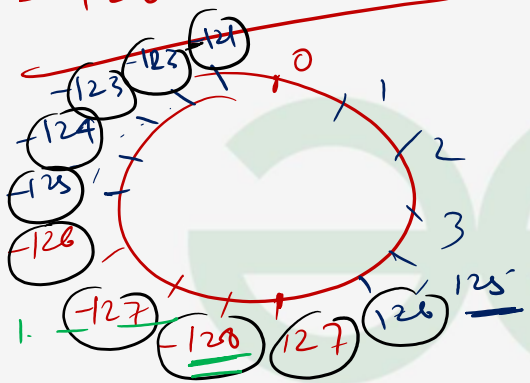
-128 to 127

Q-1)

```
main()
{
  char c = 125;
  c = c+10;
  pf("%d", c);
}
```

125 + 10 = 135

-128 + 7 = -121



Q-2)

```
main()
{
  char i;
  for(i=0; i<=127; i++)
    pf("%d", i);
}
```

Infinite loop

0, 1, 2 .. 127, -128, -127 .. -1, 0, 1 ..

Q-3)

```
main()
{
  char i;
  for(i=0; ++i; i>0)
     pf("%d", i);
}
```

o/p | 1 2 -- 127 -128 --- -1

C-Programming and DSA

Q-4)

```
main()
{
  float a=0.5, b=0.7;
  if (b<0.7)
  {
    if (a<0.5)
      Pf("Hi");
    else
      Pf("Hello");
  }
  else
    Pf("Bye");
}
```

b < 0.7

TRUE

0.7 Double

0.5 × 2 = 1.0

0 . 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0

float — Double

O/P = Hello

Size of double ≥ Size of float

b < 0.7 → Double

0.7 < 0.7

0.7 × 2 = 1.4 = 1

0.4 × 2 = 0.8 = 0

0.8 × 2 = 1.6 = 1

0.6 × 2 = 1.2 = 1

0.2 × 2 = 0.4 = 0

0.4 × 2 = 0.8 = 0

0.8 × 2 = 1.6 = 1

0.6 × 2 = 1.2 = 1

0

float

1 0 1 1 0 0 1 1 -

double

C-Programming and DSA

Control Structure:

- → if else
- → loops
- - switch

Ex:

```
main()
{ int x=5, y=10;
  if(x!=5)
      if(y==10)
          Pf("Hi");
  else
      Pf("Hello");
}
```

C-Programming and DSA

Loops:

Point to Remember

1) while(?) — Condition is Compulsory
2) for(;;)
3) for(;;);

Ex:
```
Void main()
{ int i = 1;
  while(i<=10)
  { printf("%d", i);
    i++;
  }
}
```
o/p = 1 to 10

Ex:
```
Void main()
{ int i=1;
  do
  { printf("%d", i);
    i++;
  } while(i<=10);
}
```
o/p = 1 to 10

Ex:
```
Void main()
{ int i;
  for(i=1; i<=10; printf("%d", i++));
}
```
o/p = 1 2 3 4 5 ... 10

C-Programming and DSA

Question:

```
main()
{ int x=5;
  while(x++<=7);
  printf("%d",x);
}
```

x: 5 6 7 8 9

o/p = ? (9) Ans

```
do
{ while();
}while();
```

Question:

```
main()
{ int i=1;
  do
  { while(i++<1);
  }while(i++<=2);
  printf("%d",i);
}
```

i: 1 3 4

o/p = ? (5)

Break 10 minutes

3

2

10

C-Programming and DSA

Switch():-

Syntax:

```
Switch(expression)
{
  case label1:
        body
        break;
  case label2:
        body
        break;
  :
  default: body
        break;
}
```

case 1

or 2

case e :

Switch Case :

1) All cases should be unique
2) Case Label must end with the Colon.
3) Case Label must be Constant (not variable)
4) Expression is allowed (Case 3+2-5 :)
5) Case Label should not be floating Point number
6) Empty Switch Case is allowed
7) Case and Label should have space
8) Only one default is allowed and it can be placed any where in the switch
9) Default will be executed only when no Case is matching
10) After every Case there is a requirement of break otherwise it will execute all further cases till the occurances of break or till the end of Switch
11) Default & break is not Compulsary to use
12) Continue doesn't work with switch

Ex 1:

Ex 2:

C-Programming and DSA

Ex:-

```
main()
{ int x;
  Scanf("%d", &x);
  Switch(x)
  {
    Case 0: x = x+1;
    case 1 : x = x+2;
    default : x = x+3;
  }
  printf("%d", x);
}
```

input	output
x = 0	6
x = 1	6
x = 2	5

C-Programming and DSA

Ex:-

```
main()
{
int x;
scanf("%d", &x);
switch(x)
{
case 0: x=x+1;
        break;
default: x=x-1;
case 1 : x=x+10
        break;
case 2 : x=x+100;
}
printf("%d", x);
}
```

input	output
x=0	0 1
x=1	1 11
x=2	2, 102
x=3	3, 2, 12

C-Programming and DSA

GATE:

```
main()
{ int i;
 for(i=1; i<=25; i++)
  { switch(i)
    {
      case 0: i+=5;
      case 1: i+=3;
      case 2: i+=4;
      default: i+=5;
               break;
    }
     printf("%d",i);
  }
}
```

i = 1, 4, 8, (13), 14, (19), 20, (25) 26

~~0~~ 13 19 ~~25~~ 26

C-Programming and DSA

#include <stdio.h>
void main.
{

Macro (Preprocessor Substitution) :-

1) Preprocessor will make only Substitution and it won't make any calculation

2) It will recognise those statements by (#) Symbol

3) It will be done before Compilation of Program

#

⇒

if ( )

else ( )

Ex :-

```
#include <stdio.h>
#define SQR(x) x*x
main()
{ int a;
  a = SQR(3+2);
  Pf(a); // (11) Ans
}
```

3+2*3+2,

3+6+2 = (11)

C-Programming and DSA

12 macros in C:

1) #include
2) #define
3) #ef
4) #else
5) #elif
6) #endef
7) #efdef
8) #efndef
9) #undef
10) #pragma
11) #error
12) #line

Ex:-

```
#include<stdio.h>
#if x==3
    #define y 3
#else
    #define y 7
#endif
main()
{ pf("%d",(y));
}
```

→ (7)

C-Programming and DSA

```
#define ONE 0
main()
{
#ifdef ONE
    Pf("GATE");   //GATE
#else
    Pf("CSE");
#endif
}
%p = ?
```

Void main

C-Programming and DSA

```
#define fun ma##in
        main
fun()
{
    Pf("Hi")
}
```

Hi

##

Concatenation

Basic

1) Ternary operator
2) Switch
3) Bitwise operator
4) Post Increment Decrement

Thank You !