=================== OPERATORS ========================

Evaluate the expression in a step by step way based on operator precedence and associativity

1) printf("%d\n",1==5==5); out put : zero because 1is not equal to five

2) int i =0;

printf("%d %d",i,i++); output : 1,0 because its print from right side to left

3) int x=5;

printf("%d %d %d\n",x++,x++,x++); output : 7,6,5 because its increment and printed from last element to first

4) int x=2;

printf("%d ",++x++);

printf("%d\n",x++); in above printf statement it shows error because we cant assign preincrement and post increment .

5) int k=1;

printf("%d==1 is ""%s\n",k,k==1?"TRUE":"FALSE"); its show output : true because its show k value is true ;

6) int i=5;

i=i++ - --i + ++i; output : its show 6 we evaluating from right to left 6-5+5

7) int a=7;

a+=a+=a-=6;

printf("%d\n",a); output : its show the output as 4 assignment operator is evaluating from right to left it gives output as 4

8) int x=10,y=5,p,q;

p=x>9;

q=p||(x=5,y=10);

printf("%d %d %d\n",q,x,y); output : its shows that if use or if any one statement is true its true

9) int x=2,y=1;

y+=x<<=2; after doing leftshift 2 times we get x value as 8 and after y=y+x so we get as 9

printf("%d %d\n",x,y);

10) int x=2,y=4,z;

z=y++\*x++|y--; priority is given to arthematic than unary operator

printf("%d\n",z); it gives output :13

11) int a=5,b=6,c=7,d;

d=a&=b&=c&&a; here it takes from right to left output : 0

printf("%d\n",d);

12) int i=10;

i=(10<10)?(10>=10)?(10<=10)?1:2:3:4; ternary operator is evaluated we get error

printf("%d\n",i);

13) int a=10,b=20;

a=(a>5||b=6?40:50);it will be ternary operator itwill print 50

printf("%d\n",a);

14) printf("%x\n",-1>>4); right shift operation

printf("%x\n",-1<<4); left shift operation

15) int x=7;

x=(x<<=x%2); after left shift by one we get 14

printf("%d\n",x);

16) int a=2,b=5,c=1;

printf("%d\n",(b>=a>=c?1:0)); logical operator from left to right so output :1

17) int a=5;

a=a-~a +1;

printf("%d\n",a);

18) a=b=c=1;

x=--a||++b\*(3-1)/2&&b\*(--c/3); output will be zero

printf("%d\n",x);

19) a=10

b = -5

c = 2.5 output 8 -5

printf("%d %d",sizeof(a) + sizeof((++b) \*c, b);

20) int i=5;

i=i++ - i

printf("%d",i) output : 5-6=-1

========================================= PREPROCESSORS AND MACROS ===========================

1) What will be the output of the C program?

#include<stdio.h>

#define CONDITION(i)\ \\ we define preprocessor and define macro to it

printf("preprocessor works\n");

int main()

{

CONDITION(0); we define instead of condition compiler print preprocessor work will be print

return 0;

}

2) What will be the output of the C program?

#include<stdio.h>

# define loop while(true)

int main()

{

loop; \\ it shows error because loop is not defined

printf("preprocessor-aptitude");

return 0;

} output : error

3) What will be the output of the C program?

#include<stdio.h>

# define x –5 [\\we](file:///\\we) should mentioned decrement macro to it

int main()

{

printf("%d",x); it wont be replace by decrement

return 0;

}

Output : it give error for it

4) What will be the output of the C program ?

#include<stdio.h>

#define sqr(x) ++x \* ++x \\ we define macro sqr

int main()

{

int a = 3, z; \\ we define value 3 to a

z = ++a \* ++a; here a value increment and then multiplication

a -= 3; here a =a-3;

printf("%d %d", sqr(a), z);

return 0;

}

Output : it gives 16 and 25

5) What will be the output of the C program?

#include<stdio.h>

#define x 1+2 here we have taken x is as macro variable and value are 1,2

int main()

{

int i;

i = x \* x \* x; here x value is 1 first and then second time x value as 2 so it add to macro expression so 6 +1 =7

printf("%d",i);

}

so out put : 7

6) What will be the output of the C program?

#include<stdio.h>

#define a = here we define a as macro before compiler it will save as 6

int main()

{

int num a 6;

printf("%d",num); here output will be 6

return 0;

}

7) What will be the output of the C program?

#include<stdio.h>

#define fun(x,y) x\*y

int main()

{

int x = 2, y = 1;

printf("%d",fun(x + 2, y - 1));

return 0;

}

8) What will be the output of the C program?

#include<stdio.h>

int main()

{

char DATE[15] = "Current Date";

printf("%s\n", \_\_DATE\_\_ ); here it print the current date which is predefined macro

return 0;

}

9) What will be the output of the C program?

#include<stdio.h>

int main()

{

char TIME[15] = "Current Time";

printf("%s\n",\_\_TIME\_\_); here it print the current time which is predefined macro so

return 0;

}

10) What will be the output of the C program?

#include<stdio.h>

int main()

{

printf("Line :%d\n", \_\_LINE\_\_ ); it represent current line

return 0;

}

11) What will be the output of the C program?

#include<stdio.h>

#define preprocessor\_works(x, y) \

printf(#x " and " #y " are great!\n") here #x = you and #y =me

int main(void) {

preprocessor\_works(you, me); so it print the you and me are great

return 0;

}

12) What will be the output of the C program?

#include<stdio.h>

# define puts "%s C preprocessor" the macro we assign puts = c preprocessor so it print ( preprocessor ,preprocessor )

int main()

{

printf(puts, puts);

return 0;

}

13) What will be the output of the C program?

#include<stdio.h>

#define preprocessor(n) printf ("macro" #n " = %d", macro##n) in these line n value is equal to 25 so it represent macro 25 = 47 it will give output

int main(void) {

int macro25 = 47;

preprocessor(25);

return 0;

}

14) What will be the output of the C program?

#include<stdio.h>

#include<string.h>

#define MACRO(num) ++num here we define the macro preprocessor , 13

int main()

{

char \*ptr = "preprocessor";

int num =strlen(ptr); length will be 12

printf("%s ", MACRO(ptr)); here it will preprocessor

printf("%d", MACRO(num));

return 0;

}

15) What will be the output of the C program?

#include<stdio.h>

#define i 10

int main()

{

#define i 20 here we define locally the preprocessor so it print 20

printf("%d",i);

return 0;

}

16) What will be the output of the C program?

#include<stdio.h>

#define clrscr() 50 macro define with value 50

int main()

{

clrscr();

printf("%d\n",clrscr());here it will print 50

return 0;

}

17) What will be the output of the C program?

#include<stdio.h>

#define int char

main()

{

int i=5; here we replace int by char so size of char is 1 byte

printf ("sizeof (i) =%d", sizeof (i));

}

18) Write programs to understand the usage of below preprocessor directives.

#include, #if, #ifdef, #ifndef, #else, #elif, #endif, #define, #undef, #line, #error, and #pragma

#include :

#include <stdio.h>

#define MULTIPLY(a, b) a\*b

int main()

{

// The macro is expanded as 2 + 3 \* 3 + 5, not as 5\*8

printf("%d", MULTIPLY(2+3, 3+5));

return 0;

}

**#define**: **The #define preprocessor directive is used to define constant or micro substitution. It can use any basic data type.**

**#include <stdio.h>**

**#define PI 3.14**

**main() {**

**printf("%f",PI);**

**}**

**#undef :** The #undef preprocessor directive is used to undefine the constant or macro defined by #define.

#include <stdio.h>

#define number 15

**int** square=number\*number;

#undef number

main() {

   printf("%d",square);

}

#ifdef :  The #ifdef preprocessor directive checks if macro is defined by #define. If yes, it executes the code otherwise #else code is executed, if present.

***#include <stdio.h>***

***#include <conio.h>***

***#define NOINPUT***

***void main() {***

***int a=0;***

***#ifdef NOINPUT***

***a=2;***

***#else***

***printf("Enter a:");***

***scanf("%d", &a);***

***#endif***

***printf("Value of a: %d\n", a);***

***getch();***

***}***

***#ifndef:* The #ifndef preprocessor directive checks if macro is not defined by #define. If yes, it executes the code otherwise #else code is executed, if present.**

**#include <stdio.h>**

**#include <conio.h>**

**void main() {**

**int a=0;**

**#ifndef INPUT**

**a=2;**

**#else**

**printf("Enter a:");**

**scanf("%d", &a);**

**#endif**

**printf**("Value of a: %d\n", a);

getch();

}

#if : **The #if preprocessor directive evaluates the expression or condition. If condition is true, it executes the code otherwise #elseif or #else or #endif code is executed.**

**#include <stdio.h>**

**#include <conio.h>**

**#define NUMBER 0**

**void main() {**

**#if (NUMBER==0)**

**printf("Value of Number is: %d",NUMBER);**

**#endif**

**getch();**

}

**#else :**

The #else preprocessor directive evaluates the expression or condition if condition of #if is false. It can be used with #if, #elif, #ifdef and #ifndef directives.

#include <stdio.h>

**#include <conio.h>**

**#define NUMBER 1**

**void main() {**

**#if NUMBER==0**

**printf("Value of Number is: %d",NUMBER);**

**#else**

**print("Value of Number is non-zero");**

**#endif**

**getch();**

**}**

**#error :The #error preprocessor directive indicates error. The compiler gives fatal error if #error directive is found and skips further compilation process.**

**#include<stdio.h>**

**#include<math.h>**

**#ifndef \_\_MATH\_H**

**#error First include then compile**

**#else**

**void main(){**

**float a;**

**a=sqrt(7);**

**printf("%f",a);**

**}**

#pragma : The #pragma preprocessor directive is used to provide additional information to the compiler. The #pragma directive is used by the compiler to offer machine or operating-system feature.

**#include<stdio.h>**

**#include<conio.h>**

**void func() ;**

**#pragma startup func**

**#pragma exit func**

**void main(){**

**printf("\nI am in main");**

**getch();**

**}**

**void func(){**

**printf("\nI am in func");**

**getch();**

**}**

**Elif :**

**#include <stdio.h>**

**#define YEARS\_OLD 12**

**int main()**

**{**

**#if YEARS\_OLD <= 10**

**printf("TechOnTheNet is a great resource.\n");**

**#elif YEARS\_OLD > 10**

**printf("TechOnTheNet is over %d years old.\n", YEARS\_OLD);**

**#endif**

**return 0;**

**}**

**Endlif :**

===================================== TYPEDEF =========================================

1) In the following code, the P2 is Integer Pointer or Integer?

typedef int \*ptr;

ptr p1, p2; it is pointer

2) In the following code what is 'P'?

typedef char\* charp;

const charp P; its apointer

3) What is x in the following program?

#include<stdio.h>

int main()

{

typedef char (\*(\*arrfptr[3])())[10];

arrfptr x;

return 0;

} x is an array pointer

4) What is apfArithmatics in the below statement?

typedef int (\*apfArithmatics[3])(int,int); //integer pointer

5) What is pf in the below statement?

typedef int (\*pf)(int); //pointer to function

6) What do the following declarations mean?

typedef char \*pc; pointer to char type

typedef pc fpc(); is function with return type char

typedef fpc \*pfpc; pointer to the fp()

typedef pfpc fpfpc(); is function with pointer returns

typedef fpfpc \*pfpfpc; pointer to an above function

pfpfpc a[N]: array of pointer of size N

7) Write few programs using typedef on structures and enums.

Example : structures

#include <string.h>

typedef struct Books {

char title[50];

char author[50];

char subject[100];

int book\_id;

} Book;

int main( ) {

Book book;

strcpy( book.title, "C Programming");

strcpy( book.author, "Nuha Ali");

strcpy( book.subject, "C Programming Tutorial");

book.book\_id = 6495407;

printf( "Book title : %s\n", book.title);

printf( "Book author : %s\n", book.author);

printf( "Book subject : %s\n", book.subject);

printf( "Book book\_id : %d\n", book.book\_id);

return 0;

}

Example : enum

typedef enum month billing\_cycle;

billing\_cycle month\_due;

typedef enum day {sun, mon, tue, wed, thu, fri, sat} day;

day today;

typedef struct date {

day day\_of\_week;

enum month month\_of\_year;

int day\_of\_month;

} date;

date due\_date;