1. Find the output(branching and looping)

void main()

{

int a[10] , i = 0;

for(i = 0 ; i<10 ; i++)

a[i] = 9 - i;

for(i = 0 ; i < 10 ; i++)

a[i] = a[a[i]];

for(i = 0 ; i < 10 ; i++)

printf("%d " , a[i]);

}

Output: 0 1 2 3 4 4 3 2 1 0

1. Find the output(strings)

void printout(char\* pstr)

{

int iretval = 0;

if(pstr)

{

while(\*pstr && \*pstr<= '9' && \*pstr >= '0')

{

iretval = (iretval\*10)+(\*pstr - '0');

pstr++;

}

}

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

}

void main()

{

printout("X32");

printout("47X74");

}

Output : 0

47

1. Find the output(arrays)

void main()

{

int a[] = { 8 , 9 , 9 , 9};

int r[5] = { 0 , 0 , 0 , 0 , 0};

int i = 0 , m = 1 , s = 4;

for( i = s-1 ; i >= 0 ; i--)

{

r[i+1] = (a[i] + m ) % 10;

m = (a[i] + m) /10;

}

r[0] = m ;

for( i = 1 ; i <= s ; i++)

printf("%d" , r[i]);

}

Output : 9000

1. Find the output(Functions)

int i = 0 ;

int fun(int a)

{

i++;

if( a > 99)

return a-12;

return fun(fun(a+25));

}

void main()

{

printf("%d " , fun(69));

printf("%d" , i);

}

Output : 96 7

1. Find the output(Functions)

int max(int x , int y)

{

return (y > x) ? y : x;

}

void main()

{

int a[] = { -6 , - 7 , 8 , - 9 , -2 , 3,-4,5};

int value = a[0] , ctval = a[0];

int i = 0 , n = 8;

for( i = 1 ; i < n ; i++)

{

ctval = max( a[i] , ctval+a[i]);

value = max(value , ctval);

}

printf("%d" ,value) ;

}

Output : 8

1. Find the output(operator and expression)

void main()

{

int a=30,b=40,x;

x=(a=10)&(b=50);

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

}  
output : x = 2

1. Find the output(operator and expression)

void main()

{

int x=0,y=1;

y=x;

x=!y;

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

}

Output : x = 1 y = 0

1. Find the output(operator and expression)

void main()

{

int x=3,y=4,z=4;

printf("ans=%d\n",(z>=y>=x?100:200));

}

Output : ans = 200

1. Find the output(data types)

void main()

{

float a=12.25,b=13.65;

if(a==b)

printf("a and b are equal");

else

printf("a and b are not equal");

}

Output : a and b are not equal

1. Find the output(operator and expression)

void main()

{

if('Z' < 'z')

printf("Pilots are on strike...\n");

else

printf("for absolutely outlandish demands\n");

}

Output : Pilots are on strike...

1. Find the output(operator and expressions)

void main()

{

float a = 0.7;

if( a < 0.7)

printf("Stoned");

else

printf("Avenged");

}  
output : Stoned

1. Find the output(operators and expression)

void main()

{

float a=0.5,b=0.9;

if(a&&b>0.9)

printf("tce-cse-a\n");

else

printf("tce-cse-b\n");

}

Output : tce-cse-b

Explanation: b>0.9 is evaluated first thus it gives false and a&&false

is false so the else block is executed.

1. Find the output(branching and looping)

void main()

{

int i;

for(i=1;i++<=5;printf("%d ",i));

}

Output : 2 3 4 5 6

1. Find the output(branching and looping)

void main()

{

int i = 1 , j = 1;

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

j=i++<=5;

}

Output:

2 1

3 1

4 1

5 1

6 1

7 0

1. Find the output(operators and expressions)

void main()

{

int x=3,y,z;

z=y=x;

z\*=y=x\*x;

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

}

Output : x = 3 y = 9 z =27

1. Find the output(operator and expressions)

void main()

{

int x=3,z;

z=x/++x;

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

}

Output : x = 4 z = 1

Explanation: According to operator precedence prefix/postfix executes first and then divison executes.

1. Find the output(operator and expressions)

void main()

{

int x , y , z;

x=y=z=1;

z=++x || ++y&&++z;

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

}

Output : x = 2 y = 1 z = 1

1. Find the output( branching and looping)

void main()

{

char ch='E';

switch(ch)

{

case(ch>=65 && ch<=90):

printf("Capital letter\n");

break;

case(ch>=97 && ch<=122):

printf("small letter\n");

break;

case (ch>=48 && ch<=57):

printf("Digit");

break;

default:printf("Anyother");

}

Output : error

Explanation : we cannot use any condition checking (here &&) in case. So, error is produced.

1. Find the output(branching and looping)

void main()

{

int i = 3;

switch(i)

{

case 1: printf("cse\t");

case 2: printf("It\n");break;

case 3: continue;

default : printf("goodbye");

}

}

Output : continue statement not with in a loop

1. Find the output(branching and looping)

void main()

{

char s;

switch(s)

{

case '1': printf("database");

case '2': printf("data-structure");

default: printf("c");

printf("byebye");

}

}

Output: cbyebye

1. Find the output(branching and looping)

void main()

{

int k=-2,j=4;

switch(k/=j/k)

{

default:printf("lenovo");

case 0 : printf("hp");

case 1: printf("acer");

case 2: printf("dell");

}

}

Output : acerdell

1. Find the output(branching and looping)

void main()

{

int j,x=0;

for (j=0;j<=5;j++)

{

switch(j-1)

{

case 0:

case -1:

x -= 1;break;

case 1:

case 2:

case 3:

break;

default: x+=3;

}

printf("%d " , x);

}

}

Output : -1 -2 -2 -2 -2 1

1. Find the output(branching and looping)

void main()

{

int i;

for(i = 2 ; i <= 10 ; i++)

{

switch(i)

{

case 2: printf("0");continue;

case 3: break;

case 4:

case 5:printf("1");break;

default: printf("000");

}

}

}

Output : 011000000000000000

1. Find the output (branching and looping)

void main()

{

char ch='E';

switch(ch) {

case(ch>=65 &&ch<=90):

printf("Capital letter\n");

break;

case(ch>=97 &&ch<=122):

printf("small letter\n");

break;

case (ch>=48&&ch<=57):

printf("Digit");

break;

default: printf("Any other");

}

}

Output : error

1. Find the output(branching and looping)

void main()

{

int i , j ;

for(j=1;j<=10;j++)

for(i=1;i<=10;i++)

if(j<10)

{

printf("tce mit psg");

}

else

printf("anna university");

}

Output :

1. Find the output(functions)

int funcl(int k)

{

k++;

return k;

}

void main()

{

int k=35,z;

k=funcl(k=funcl(k=funcl(k)));

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

}

Output : k = 38

1. Find the output(functions)

void pri(int,int);

void printit(float,int);

void main()

{

float a=3.14;

int i=99;

pri(i,a);

printit(a,i);

}

void pri(int i,int a)

{

printf("i=%d a=%f\n",i,a);

printf("a=%f i=%d\n",a,i);

}

void printit(float a,int i)

{

printf("a=%f i=%d\n",a,i);

printf("i=%d a=%f\n",i,a);

}

Output :i=99 a = 0.000000

a=0.000000 i=0

a=3.140000 i=99

i=99 a=3.140000

1. Find the output(operator and expression)

void main(){

int k=35,\*z,\*y;

z=&k; y=z;

\*z++=\*y++;

k++;

printf("k=%d z=%d y=%d",k,z,y);

}

Output : k = 36 z = addressofk y = addressofk here z and y points to same address that is the address of z

1. Find the output(pointer)

void main(){

int a=100,\*b,\*\*c,\*\*\*d;

b=&a; c=&b; d=&c;

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

}

Output : 100 100 100 100

1. Find the output(operator and expression)

void main(){

int z=40;

printf("%d\n",printf("%d,%d\n",z, z-1));

}

Output :

40,39

6

EXPLANATION: It prints the number of characters in the output. Here , and \n is also considered as a character. So 4, 0, , , 3, 9, \n totally 6 characters are there.

1. Find the output( functions)

void junk(int,int\*);

int main()

{

int i=-5,j=-2;

junk(i,&j);

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

return 0;

}

void junk(int i,int \*j)

{

i=i\*1;

\*j=\*j \* i;

}

Output : i=-5 j=10

1. Find the output( pointers)

float \*jam(float \*r){

r=r+1;

return (r);

}

void main()

{

float \*jam(float \*);

float p=23.5,\*q;

q=&p;

printf("q before call=%d\n",q);

q=jam(&p);

printf("q after call=%d",q);

}

Output : q before call 2293528

Q after call 2293532

1. Find the output(function)

void main()

{

int i;

printf("hai");

for(i = 1 ; i<= 10 ; i++)

main();

}

Output: recursive infinite function call

1. Find the output(branching and looping)

void main()

{

if(printf("C for yourself how it works\n"))

main();

}

Output : recursive call infinite times

1. Find the output(datatypes)

void main()

{

unsigned int ch=0;

for(ch=65;ch<=255;)

printf("%d %c\n",ch,ch++); // STACK(right to left it is executed) ch value will be the finally updated value of ch

}

Output : 66 A

67 B

68 C

69 D

70 E etc….

Explanation: print next ascii value followed by corresponding characters this is because the execution is from right to left like that in stack.

1. Find the output( datatypes)

void main()

{

float a=0.7;

double b=0.7;

if(a==b)

printf("condition statisfied");

else

printf("condition not statisfied");

printf("\na=%f b=%lf\n",a,b);

}

Output : condition not statisfied

A=0.700000 b=0.700000

1. Find the output(datatypes)

void main()

{

float y=0.9;

long double z=0.9;

if(y=-z)

printf("icecrearm");

else

printf("cake");

}

Output: icecream

1. Find the output(datatypes)

void change()

{

auto int i=100;

register int j=200;

printf("change's i and j are %d %d\n",i,j);

}

void main()

{

auto int i=10;

register int j=20;

printf("main's I and j are %d %d\n",i,j);

change();

printf("main's I and j are %d %d\n",i,j);

}

Output:

Main’s I and j are 10 20

Change’s I and j are 100 200

Main’s I and j are 10 20

1. Find the output( operator and expression)

void main()

{

double x,d=4.4;

int i=2,y;

x=(y=d/i)\*2;

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

y=(x=d/i)\*2;

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

}

Output:x=4.000000 y=2

X=2.200000 y=4

1. Find the output(operator and expression)

void main()

{

double x,d=5.0;

int y;

x=d\*(x=2.5/d);

printf("x=%lf\n",x);

x=d\*(y=(int)2.5+1.5);

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

}

Output : x=2.500000

x=15.000000 y=3

1. Find the output(function)

void main()

{

int c=5;

printf("c=%d\n",c--);

if(c) main();

}

Output : c=5 is getting printed infinite times

1. Find the output(functions)

int func(int x)

{

static int v=2;

v--;

return (v-x);

}

int i;

void main()

{

int j;

for(;;)

{

if( j= func(i) )

printf("j= %d " , j);

else

break;

}

}

Output : j=1

1. Find the output(data types)

void main()

{

long num=2;

short n=2;

signed no=2;

printf("num=%ld n=%d no=%d\n",num,n,no);

}

Output : num=2 n=2 no=2

1. Find the output(datatypes)

void main()

{

char ch=122,ch1='z';

printf("ch=%c\n",ch);

printf("chl=%d\n",ch1);

}

Output:ch = z

Ch1 = 122

1. Find the output(datatypes)

void main()

{

unsigned int a=25;

unsigned b=25;

long unsigned c=345L;

long signed d=345L;

printf("a=%u b=%u\n",a,b);

printf("c=%lu d=%d\n",c,d);

}

Output:a=25 b=25

c=345 d=345

1. Find the output(datatypes)

void main()

{

auto int i=100;

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

i+=1;

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

}

Output: i=100

I=101

1. Find the output(data types)

#p1

void main()

{

register int i;

for(i=1;i<=100;i++)

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

}

#p2

void main()

{

auto int i;

for(i=1;i<=100;i++)

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

}

Output : both programs print 1 to 100

1. Find the output ( operator and expression)

#define CUBE(x) x\*x\*x

void main()

{

int a;

a= 27 / CUBE(3);

printf("%d" , a);

}

Output : 81

1. Find the output(operator and expression)

#define CUBE(x) (x\*x\*x)

void main()

{

int a , b=2;

a = CUBE(b+4) / b++;

printf("a= %d b = %d ", a , b);

}

Output : a= 11 b =3

Explanation: (2+4\*2+4\*2+4)/2 => (2+8+8+4)/2=> 22/2 => 11 and b becomes 3

1. Find the output(operator and expression)

#define AND &&

#define OR ||

#define LE <=

#define GE >=

void main()

{

char ch='D';

if((ch GE 65 AND ch LE 90) OR (ch GE 97 AND ch LE 122))

printf("Alphabet\n");

else

printf("Not alnhabet");

}

Output : Alphabet

1. Find the output( arrays)

void main()

{

static float arr[]={1.2,12,2.4,24,3.5,35};

int i;

for(i=0;i<=5;i++)

printf("%f ",arr[i]);

}

Output:1.200000 12.000000 2.400000 24.00000 3.500000 35.000000

1. Find the output(arrays)

void main()

{

static int b[]={10,20,30,40,50};

int i;

for(i = 0; i<= 4 ; i++)

printf("%d ",b[i]);

}

Output : 10 20 30 40 50

1. Find the output(arrays)

void main()

{

static int a[5]={5,10,15,20,25};

int i,j,m,n;

i=4-a[1];

j=a[1]++;

printf("i=%d j=%d a[1]=%d\n",i,j,a[1]);

i=1;

m=a[1]+41;

printf("i =64 m=%d\n",i,m);

i=2;

n=a[1]++;

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

}

Output : i=-6 j=10 a[1]=11

i=64 m=1

i=2 n=11

1. Find the output(arrays)

void main()

{

static int a[]={10,20,30,40,50};

int j;

for (j=0;j<5;j++)

{

printf("%cl\n",\*a);

a++;

}

Output : lvalue required error

1. Find the output(arrays)

void main()

{

static int b[]={10,20,30,40,50};

int i,\*k;

k= &b[4]-4;

for(i=0;i<=4;i++)

printf("%d ",\*k);

k++;

}

Output:10 10 10 10 10

EXPLANATION: Since there is no brackets given printf(“%d”.\*k); this line alone comes under the scope of the for loop. The line k++; executes after the for loop is fully executed

1. Find the output(arrays)

void main()

{

static int a[]={2,4,6,8,10};

int i;

for(i=0;i<=4;i++)

\*(a+i)=a[i]+i[a];

printf("%d\n",\*(i+a));

}

Output : -1

1. Find the output(pointers)

void main()

{

int arr[]={0,1,2,3,4};

int i,\*ptr;

for(ptr=&arr[0],i=0;i<=4;i++)

printf("%d ",ptr[i]);

}

Output: 0 1 2 3 4

1. Find the output(pointers)

void main()

{

int arr[]={0,1,2,3,4};

int i,\*p;

for(p=arr,i=0;p+i<=arr+4;p++,i++)

printf("%d ",\*(p+i));

}

Output : 0 2 4

1. Find the output(pointer)

void main()

{

int arr[]={0,1,2,3,4};

int i,\*ptr;

for(ptr=arr+4;ptr>=arr;ptr--)

printf("%d ",\*ptr);

}

Output : 4 3 2 1 0

1. Find the output(pointer)

void main()

{

int arr[]={0,1,2,3,4};

int i,\*ptr;

for(ptr =arr+4,i=0;i<=4;i++)

printf("%d ",ptr[-i]);

}

Output : 4 3 2 1 0

1. Find the output(pointer)

void main()

{

int arr[]={0,1,2,3,4};

int \*ptr,i;

for(ptr=arr+4;ptr >= arr ; ptr--)

printf("%d ",\*ptr);

}

Output : 4 3 2 1 0

1. Find the output(pointer)

void main()

{

static int a[]={0,1,2,3,4};

static int \*p[]={a,a+1,a+2,a+3,a+4};

int \*\*ptr=p;

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

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

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

}

Output:

addressofa 0

addressofp addressofa 0

addressofp addressofa 0

1. Find the output(pointer)

void main()

{

static int a[]={0,1,2,3,4};

static int \*p[]={a,a+1,a+2,a+3,a+4};

int \*\*ptr=p;

printf("%d %d %d\n",ptr-p,\*ptr-a,\*\*ptr);

\*ptr++;

printf("%d %d %d\n",ptr-p,\*ptr-a,\*\*ptr);

\*++ptr;

printf("%d %d %d\n",ptr-p,\*ptr-a,\*\*ptr);

++\*ptr;

printf("%d %d %d\n",ptr-p,\*ptr-a,\*\*ptr);

}

Output : 0 0 0

1 1 1

2 2 2

2 3 3

1. Find the output(pointer)

void main()

{

static int a[]={0,1,2,3,4};

static int \*p[]={a,a+1,a+2,a+3,a+4};

int \*\*ptr=p;

\*\*ptr++;

printf("%d %d %d\n",ptr-p,\*ptr-a,\*\*ptr);

\*\*++ptr;

printf("%d %d %d\n",ptr-p,\*ptr-a,\*\*ptr);

++\*\*ptr;

printf("%d %d %d\n",ptr-p,\*ptr-a,\*\*ptr);

}

Output:

1 1 1

2 2 2

2 2 3

1. Find the output(arrays)

void main()

{

static int n[3][3]={12,4,3,6,8,5,3,5,11};

printf("%d %d %d\n",n,n[2],n[2][2]);

}

Output : startingaddress 2rowstartingaddress 11

1. Find the output(strings)

void main()

{

char s[]="Rendezvous !";

printf("%d\n",\*(s+strlen(s)));

}

Output : 0

1. Find the output(Arrays)

void main()

{

char str[20];

static int i;

for(;;) {

i++[str]='A'+2;

if(i==19)

break;

}

i[str]=0;

printf("%s" , str);

}

Output : CCCCCCCCCCCCCCCCCCC

1. Find the output( strings)

void main()

{

char s[]="C smart!!";

int i;

for(i=0;s[i];i++)

printf("%c%c%c%c\n",s[i],\*(s+i),i[s],\*(i+s));

}

Output :

CCCC

ssss

mmmm

aaaa

rrrr

tttt

1. Find the output(strings)

void main()

{

char s[]="Dinks Grunts and Guffaws";

printf("%c\n",\*(&s[2]));

printf("%s\n",(s+5));

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

printf("%c\n",\*(s+10));

}

Output :

n

Grunts and Guffaws

Dinks Grunts and Guffaws

t

1. Find the output(strings)

void main()

{

char str[]="MalayalaM";

char \*s;

s = str+8;

while( s > str)

{

printf("%c" , \*s);

s--;

}

}

Output: Malayala

1. Find the output(strings)

void main()

{

char str[]="Shall we tell the Deputy Director?";

printf("%s\n%s\n%s\n",str,str+6,str+9);

}

Output :

Shall we tell the Deputy Director?

we tell the Deputy Director?

tell the Deputy Director?

1. Find the output(structures and union)

struct employee

{

char name[25];

int age;

float bs;

};

void main()

{

struct employee e;

e.name = "Hacker";

e.age=25;

printf("%s %d" , e.name , e.age);

}

Output :error

Explanation: strcpy(e.name,”Hacker”); should be used to store string

1. Find the output(structures and union)

struct name1

{

char name[25];

char lang[10];

};

static struct name1 a = {"Hacker" , "cr"};

void main()

{

printf("%s %s" , a.name , a.lang);

}

Output : Hacker cr

1. Find the output(structures and union)

struct a {

char ch[7];

char \*str;

};

void main()

{

static struct a s1={"Nagpur" , "Bombay"};

printf("%c %c\n" , s1.ch[0] , \*s1.str);

printf("%s %s" , s1.ch , s1.str);

}

Output :

N B

Nagpur Bombay

1. Find the output(structures and union)

struct a

{

int i;

char ch[4];

};

union b

{

int j;

char ch[4];

};

void main()

{

printf("%d " , sizeof(struct a));

printf("%d " , sizeof(union b));

}

Output : 8 4

1. Find the output(structures and union)

union a

{

int i;

char ch[2];

};

void main()

{

union a u;

u.i = 256;

printf("%d %d %d " , u.i , u.ch[0] , u.ch[1]);

}

Output : 256 0 1

Explanation: The binary representation of 256 is **00000000 00000001 00000000.**

**Ch[0] contains 00000000 so ch[0] is 0**

**Ch[1] contains 00000001 so ch[1] is 1**

1. Find the output(structures and union)

struct a

{

long int i;

char ch[4];

};

void main()

{

struct a s;

s.i = 512;

printf("%d %d %d" , s.ch[0] , s.ch[1] ,s.ch[3]);

}

Output : 2 0 0

1. Find the output(structures and unions)

union a

{

int i;

char ch[4];

};

void main()

{

union a u;

u.ch[0]=3;

u.ch[1]=2;

u.ch[2]=0;

u.ch[3]=0;

printf("%d %d %d",u.ch[0],u.ch[1], u.i);

}

Ouput : 3 2 515

Explanation: By assigning **u.ch[0] = 3** and **u.ch[1] = 2**, we set the least significant byte to **3** and the second least significant byte to **2**

Since integers are typically stored in little-endian format on most systems, the bytes are interpreted in reverse order when converting back to an integer

So, the value of **u.i** is **2\*256^1 + 3\*256^0 = 515**

1. Find the output(data types)

void main()

{

float a=3.14;

printf("a=%f\n",a);

printf("a=%6.2f\n",a);

printf("a=%-6.2\n",a);

printf("a=%6.1f\n",a);

printf("a=%6.0f\n",a);

}

Output:

a=3.140000

a= 3.14

a=

a= 3.1

a= 3

1. Find the output(strings)

void main()

{

printf("%20\s\n","short leg");

printf("%20\s\n","long leg");

printf("%20\s\n","deep fine leg");

printf("%20\s\n","backward short leg");

printf("%20\s","legs are the same");

}

Output :

short leg

long leg

deep fine leg

backward short leg

legs are the same

1. Find the output(functions)

void main()

{

printf("Hello\nHi\n");

printf("Hello\rHi\n");

printf("Hello\b\b\b\b\b\n");

printf("Hil\b\b\bBye\n");

}

Output :

Hello

Hi

Hillo

Hello

Byelo

1. Find the output(functions)

void main()

{

printf("I\tam\ta\tboy\n");

}

Output :I am a boy

1. Find the output(strings)

void main()

{

char name[20]="Sandeep";

int salary=1500;

printf("%s %d\n", name , salary);

fprintf(stdout , "%s%d\n",name,salary);

}

Output :

Sandeep 1500

Sandeep1500

1. Find the output(pointer)

void main()

{

static char str[]="Triplet";

char \*s;

s = str;

while(\*s)

{

putc(\*s , stdout);

fputchar(\*s);

printf("%c\n ",\*s);

s++;

}

}

Output:

TTT

rrr

iii

ppp

lll

eee

ttt

1. Find the output(command line arguments)

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

{

printf("%d ", argc);

printf("%s" , argv[0]);

}

Output : 1 demo.exe

1. Find the output( operator and expression)

void main()

{

short int k;

k = -35;

printf("k=%d " , k);

k = -k;

printf("k = %d " , k);

}

Output : k=-35 k=35

1. Find the output(operator and expressions)

void main()

{

int i=32,j=65,k;

k=j\*32;

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

k = j<<2; // LEFT SHIFT: x<<n = x\*(2^n)

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

k=i>>5; // RIGHT SHIFT: x>>n = x/(2^n)

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

}

Output:

K=2080

K=260

K=1

1. Find the output(operator and expressions)

void main()

{

int a=0Xff;

if(a<<4>>12)

printf("leftest");

else

printf("rightest");

}

Output : rightest

1. Find the output(data types)

void main()

{

enum status {low,medium,high};

enum status rain;

rain = 0;

if(rain == low)

printf("rain = %d", rain);

}

Output : rain =0

1. Find the output(structures and unions)

typedef struct

{

char name[20];

int age;

}a;

void main()

{

a temp= {"sunil" , 30};

printf("%s %d" , temp.name , temp.age);

}

Output : sunil 30

1. Find the output(data types)

void main()

{

printf("%f\n" , (float)(int)(float)(int)6.5/2+3.5);

}

Output : 6.500000

Explanation: first brackets is executed and then division and addition

1. Find the output(structures and unions)

struct num

{

unsigned bit0:1;

unsigned bit1:1;

unsigned bit2:1;

unsigned rest:5;

};

union a

{

struct num n;

char ch;

}b;

void main()

{

b.ch = 32;

printf("%d %d %d %d", b.n.bit0 , b.n.bit1,b.n.bit2,b.n.rest);

}

Output: 0 0 0 4

Explanation: 32 is represented as 00100000 and by following little-endian representation bit0 takes the last 1-bit and bit1 takes the one bit before bit0…..so on(32=>00100000 where 00100-rest, 0-bit2, 0-bit1, 0-bit0).

1. Find the output(functions)

int show();

void main()

{

int (\*f)();

f= show;

printf("address= %d\n",f);

}

int show()

{

printf("Diamonds are very costly");

}

Output:address = 4199264(Memory address of show)

1. Find the output(operators and expressions)

void main()

{

int a = 3 , b = 2 , c = 1 , d;

d = a | b & c;

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

d = a+ b & -c;

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

}

Output : d = 3 d = 5

EXPLANATION: Frist & then ^ then | (OPERATOR PRECEDENCE)

95.

void main()

{

float a=0.7;

double b=0.7;

if(b==0.7)

printf("condition statisfied");

else

printf("condition not statisfied");

printf("\na=%f b=%lf\n",a,b);

}

OUTPUT:

condition statisfied

a=0.700000 b=0.700000

96.

void main()

{

float a=0.7;

double b=0.7;

if(a==b)

printf("condition statisfied");

else

printf("condition not statisfied");

printf("\na=%f b=%lf\n",a,b);

}

OUTPUT: condition not statisfied

a=0.700000 b=0.700000

97.

void main()

{

float a=0.7;

double b=0.7;

if(a==0.7f) // (OR) if(a==0.7F)

printf("condition statisfied");

else

printf("condition not statisfied");

printf("\na=%f b=%lf\n",a,b);

}

OUTPUT: condition statisfied

a=0.700000 b=0.700000

98.

void main()

{

float a=0.7;

double b=0.7;

if(a < 0.7)

printf("condition statisfied");

else

printf("condition not statisfied");

printf("\na=%f b=%lf\n",a,b);

}

OUTPUT: condition statisfied

a=0.700000 b=0.700000

EXPLANATION: float 0.7 is lesser than 0.7

99.

void main()

{

float a=0.7;

double b=0.7;

if(b < 0.7)

printf("condition statisfied");

else if(b>0.7)

printf("condition not statisfied");

else

printf("final condition");

printf("\na=%f b=%lf\n",a,b);

}

OUTPUT: final condition

a=0.700000 b=0.700000

EXPLANATION: double 0.7 is equal to 0.7

CONCLUSION:

Float 0.7 < 0.7

Float 0.7 == 0.7f

Double 0.7 == 0.7

Float 0.7 != Double 0.7