1. Find the output for the following programs(branching and looping)

```
#include<stdio.h>
Void  main()
{
int i;
for( i = 1 ; i < 4 ; i++)
{
  switch(i)
  {
    case 1 : printf("%d" , i);break;
    case 2 : printf("%d" , i);break;
    case 3 : printf("%d" , i);break;
  }

}
switch(i)
{
    case 4 : printf("%d" , i);break;
}
}
```

   Output : 1234

   explanation : loop 3 time execute aagum athula i vanthu 1 la irunthtu 3 varaikkum swich la pogum athula i la 123 nu print aagum .but loop break aagum pothu i vanthu 4 aagi irukum so next switch case success aagi 4 um print aagum.

2. Find the output(operartor and expression)

```
void main()
{
char *s = "\12345s\n";
printf("%d" , sizeof(s));
}
```

   Output : 4

   explanation: ithula s la irukra value pakka theva illa because namba print panrathu sizeof(s) so pointers oda size 4 byte so 4 nu print aagum.

3. Find the output(Funtions)

```
int main()
{
static int i = 3;
printf("%d" , i--);
return i>0 ? main() : 0 ;
}
```

Output : 321

explanation : initial la i oda value 3 .first time 3 value 3 print aagum aprm i-- aagi i=2 aagum ithula 2>0 condition true so again main function ah call pannum ippo 2 print aagum aprm i-- aagi i=1 aagum 1>0 condition true so again main function ah call pannum ippo 1 print aagum i= 0 aagidum ippo 0>0 conditon false so program exit aagidum.

4. Find the output(pointers)

```
int main()
{
char *s[]={ "dharmr'a","hewlett-packard","siemens","ibm"};
char **p;
p = s ;

printf("%s" ,++*p);
printf("%s",*p++); ;
printf("%s" ,++*p);
}
```

Output: harmr’aharmr’aewlett-packard

exaplanation: input vanthu array of character pointers so oru oru word um array va irukum so p=s apdi assign pannum pothu “dharmr’a” oda address vanthu p point pannum ithula *p print panna “dharmr’a” print aagum aana namba pre - increment panrathunala p “dharmr’a” la irukra h ah point pannum so print panna “harmr’a” nu print aagum aprm next line post-increment so first p point pannitu irukra “harmr’a” print aagitu array increment aagidum so ippo p “hewlett-packard” point pannum ippo p pre-increment aagurathunala p inrement aagi e character point pannum ippo print pannum pothu “ewlett-packard” print aagum.

5. Find the output(dynamic memory)

```
#include<stdio.h>
#include<malloc.h>
#include<string.h>
int main()
{
int i;
char a[]="String";
char *p = "New String";
char *temp;
temp = malloc(strlen(p) + 1);
p = malloc( strlen(temp) + 1);
strcpy(p , temp);
printf("%s" , p);
}
```

Output : unpredictable string

explanation : malloc na memory allocation so temp ku memory allocate panrom evlo na p length 10 and 10+1 =11 so 11 byte create aagum aprm p kum temp oda length 0 so 0+1=1 p= 1 byte create aagum strcpy na string copy so p 1 byte la temp 11 byte ah copy panrom temp vanthu empty ah irukrathunaala garbage value varum athanala unpredictable string.

6. Find the output(algorithm)

```
int main()
{
int n = 12 , res = 1;
while( n > 3)
{
   n -= 3;
   res *= 3;
}
printf("%d" , n*res);
}
```

Output : 81

explanation:initial n=12 12>3 so n=12-3; n=9 aagidum res=1*3=3 aagidum again 9>3 condition true n=9-3=6 aagum res=3*3=9 aagum again 6>3 condition true n=6-3=3; res=9*3=27 again 3>3 condition false exit aagum print pannum pothu n and res multiple pannum pothu 3*27 aagum so ans is 81.

7. Find the output(function)

```
void fun(int [][3]);
int main()
{
int a[3][3] = {9,8,7,6,5,4,3,2,1};
fun(a);
printf("%d\n" , a[2][1]);
}
void fun(int b[][3])
{
   ++b;
   b[1][1]=5;
}
```

Output : 5
expalanation:iniially 2 D array {9,8,7
                                              6,5,4
                                              3,2,1 };
ithu 2D array a first 1 D array ahavahu {9.8.7} oda address ah point pannitu irukum itha pass panni pre-increment panna (++b) next {6,5,4} oda address ah point pannum so ippo array epdi irukum na b[2][3] so ithula b[1][1] na 2 va 5 ah mathum athvathu a array la a[2][1] oda address   so main function la a[2][1] ah print panna ah nu print aagum.

8. Find the output(strings)
```
void main()
{

   int i , n;
   char x[5];
   strcpy( x , "Zoho");
   n = strlen(x);
   *x = *(x+(n-1));
   printf("%s" , x);
}
```
Output: ooho
explanation: strcpy pannum pothu x[0]=z , x[1]=0,x[2]=h,x[3]=o irukum. Next n =strlen(x) pannum pothu x=4 length of string x next line x base address ah point pannum so n-1 na 3 .base address + 3 so x oda last o va base address athavathu z irukra address ku update pannidum so x ah print panna ah ooho nu print aagum.

9. Find the output(arrays)
```
void main()
{
   int c[]={5,4,3,4,5};
   int j , *q = c;
   for( j = 0 ; j<5 ; j++){
        printf("%d" , *c);
        ++q;
   }
}
```
Output:55555

explanation: character array c atha oru q apdinra pointer ku copy panrom c vanthu array oda base address athavathu 5 va point pannitu irukum so print panna 5 time um 55555 tha print aagum because namba copy panna q variable la tha increment panrom athu c ah affict pannathu.

10. Find the output(branching and looping)

```
void main()
{
   int i = 1;
   for(i =0 ; i= -1 ; i=1){
        printf("%d", i);
   if(i!= 1) break;
   }
}
```

Output: -1

explanation: initial i=1 after loop initialization now i=0 after condition checking i=-1 so print pannum pothu -1 print aagum if la check pannum pothu -1!=1 condition true so break aagi exit aagidum.

11. Find the output(Arrays)

```
void main()
{

   int s[] = {1,0,5,0,10,0};
   int f[] = {2,4,6,8,10,12};
   int n = 6 , i = 0 , j = 0;
   for( j = 1 ; j < n ; j++)
   {
      if( s[j] >= f[i])
      {
         printf("%d" , i);
         i = j;
      }
   }
}
```

output : 02

explanation:first time 0>2 condition false ethuyum aagathu next 5>2 true 0 print aagum i=2 aagidum next 0>6 false next 10>6 true 2 print aagum i=4 now 0>10 false next length condition false loop terminate aagidum so 02 nu print aagum.

12. Find the output(Functions)

```
void f(int *a , int m)
```

```
{
   int j = 0;
   for(j = 0 ; j < m ; j++)
   {
      *(a+j) = *(a+j) - 5;
   }
}
void main()
{
   int a[] = {'f' , 'g' , 'h' , 'i' , 'j'};
   int j = 0 ;
   f(a , 5);
   for(j = 0 ; j<= 4 ; j++)
      printf("%c\t" , a[j]);

}
```

Output:a	b	c	d	e

explanation:a array and 5 va function first time a array la first character f ah edukum f-5 pannum pothu ascii value f ku 102-5=97 athavathu a atha a[0]=a store panrom next g-5 = b atha a[1] = b nu store panrom nect h-5=c athu a[2]=c nu store aagum ithe mathiri innum 2 character num pannanum next print panna a b c d e apdinu print aagum.

13. Find the output(branching and looping)

```
void main()
{
   int i=0, j=0 , sum=0;
   for(i= 1; i < 500 ; i*=3)
     for(j=0;j<i;j++)
       sum++;
   printf("%d",sum);

}
```

Output: 364

explanation: initial sum=0 first time j=0 i=1 so second loop oda condition padi true sum=1 aagum next j=1 aagi condition false aagum ithu mathiri i time j loop execute aagum ithula mukkiyama i=i*3 aaguthu atha note pannaum so second time i=3 and 9,27,81,…

14. Find the output(branching and looping)

```
void main()
```

```
{
    int n;
    for(n = 6 ; n!= 1; n--)
         printf("%d" , n--);
}
```

Output: infinite loop

n vanthu decrement 2 times nadakkuthu so athu even aavetha irukume thavira epppathu 1 aaga chance illa so infinite loop.

15. Find the output(arrays)

```
void main()
{
    int a[3][4] = {2,4,6,5,10,12,12,10,5,6,4,2};
    int i = 0 , j , k =99;
    while(i < 3)
    {
        for(j = 0 ; j < 4 ; j= j++)
        {
            if( a[i][j] < k)
            {
                k = a[i][j];
            }
        }
        i++;
    }
    printf("%d" , k);
}
```

Output : 2

```
explanation: a={2,4,6,5
                10,12,12,10
                5,6,4,2
               };
```

outer while loop 3 time and inner for loop 4 time execute aagum and k=99 first 2<99 true so k=2 nu aagidum next 4<2 false 6<2 and 5 <2 false so ippo i++ aagum so array la patha 2 tha minimum athanala inimel k chance aaga vaippu illa so 2 nu print aagum.

16. Find the output(pointer)

```
void main()
{

char *x="Alice";
int i , n = strlen(x);
*x = x[n];
for(i=0; i<=n; i++)
{
printf("%s ", x); x++;
printf("\n", x);
}

return 0;


}
```

Output : runtime error

n=5 so x[5] vathu null irukum because 0 to 4 varikkum tha x la values irukum 5 null ah irukum so null x ah increment panrathu naala runtime error.

17. Find the output(structures and union)

```
struct value{
int bit1:1;
int bit3:4;
int bit4:4;
}bit;
int main()
{
printf("%d\n", sizeof(bit));
return 0;
}
```

Output : 4

explanation: ithula

18. Find the output(dynamic memory)

```
struct node
{
int data;
float d;
struct node *link;
};
int main()
```

```
{
struct node *p, *q;
p = (struct node *) malloc(sizeof(struct node));
q = (struct node *) malloc(sizeof(struct node));
printf("%d, %d\n", sizeof(p), sizeof(q));
return 0;
}
```

Output : 8 , 8

Explanation :

→Depends on complier

19. Find the output(structures and unions)

```
typedef union
{
    int a;
    char b[10];
    float c;
}Union;
int main()
{
    Union x , y = {100};
    x.a = 50;
    strcpy(x.b , "Hello");
    x.c = 21.50;
    printf("%d %s %f\n" , x.a , x.b , x.c);
    printf("%d %s %f" , y.a,y.b, y.c);

}
```

```
Output:1101791232  21.500000
100 d 0.000000
```

20. Find the output(structures and union)

```
struct point{

int x;
int y ;
};
struct point origin , *pp;
int main()
{
    pp = &origin;
    printf("origin is (%d %d)\n", (*pp).x , (*pp).y);
```

```
   printf("origin is (%d %d)" , pp->x ,  pp->y);
   return 0;
}
```

Output : origin is (0 0)
origin is (0 0)
Explanation :
→This is a very good example for accessing structure variables

21. Find the output(branching and looping)
```
void main()
{
int i = -1;
printf("i =%d +i = %d\n" , i , +1);
}
```
Output : i=-1 i=1
Explanation :
→here the i value is not incremented and printed
→ here +1 means explicitly positive 1 So, it prints 1.

22. Find the output(datatypes)
```
void main()
{
char not;
not=12;
printf("%d",not);

}
```
Output : 12

23. Find the output(branching and looping)
```
#define FALSE -1
#define TRUE 1
#define NULL 0
void main()
{
if(NULL)
 puts("NULL");
else if(FALSE)
 puts("TRUE");
else
 puts(" FALSE");
}
```

Output : TRUE

Explanation :

→ The #define is called macro definitions i.e , FALSE = = -1

→first if statement is false as NULL == 0 (macro definition)

→else if statement is true(as any non-zero number is consider as true in c)

24. Find the output(operator and expressions)

```
void main()
{
   int k = 1;
   printf("%d==1 is"" %s" ,k, k == 1 ? "TRUE":"FALSE");
}
```

Output : 1==1 is TRUE

25. Find the output(file manipulation)

```
int main()
{
FILE *ptr;
char i;
ptr=fopen("demo.c","r");
while((i=fgetch(ptr))!=EOF)
printf("%c",i);
}
```

26. Find the output(branching and looping)

```
int main()
{
int t , i ;
for ( t=4;scanf("%d",&i)-t;printf("%d\n",i))
   printf("%d--",t--);

}
```

Output : loop runs 4 times

27. Find the output(structures and unions)

```
struct emp{
int len;
char name[1];
};
int main()
{
char newname[] = "Rahul";
struct emp *p = (struct emp *) malloc(sizeof(struct emp) -1 + strlen(newname)+
1);
p->len = strlen(newname);
strcpy(p -> name, newname);
printf("%d %s\n", p->len, p->name); return 0;
}
```

Output : 5 Rahul

Explanation :

→ Let's see deep into memory allocation

→if you check the sizeof(struct emp) we get 8(but as per logic we need to get 5 as output this is because the compiler adds 3 padding bits for performance).

→ So here we are assigning p->len to strlen(newname) which is 5.

→Using (String copy)strcpy function we copy the value in newname to p->name.

Bonus point:

→ if you want to allocate without padding bits use `#pragma pack(1)` `header(caution it may reduce performance).`

28. Find the output(algorithm)

```
int main() {
printf("%d %d %d %d\n",72,072,0x72,0X72);
return 0;
}
```

Output : 72 58 114 114

Explanation :

→072 here 0 means octal number then it is converted to binary then to integer so output is 58.

→0x72 & 0X72 here x and X both mean hexadecimal then convert it to binary then to decimal.

29. Find the output(operator and expression)

```
void main()
{
 char ch;
 int a;
 float b;
printf("bytes occupied by ch=%d\n",sizeof(ch));
printf("bytes occupied by a=%d\n",sizeof(a));
printf("bytes occupied by b=%d\n",sizeof(b));
}
```

Output :

```
Bytes occupied by ch=1
Bytes occupied by a=4
Bytes occupied by b=4
```

30. Find the output(operator and expressions)

```
void main()
{
    printf("%d\t" , sizeof('7'));
    printf("%d\t" , sizeof(7));
    printf("%d" , sizeof(7.0));
}
```
Output: 4 4 8

Explanation :

→we are checking the size('7') which is character output should be 1byte but its 4 byte because we are printing it as integer(%d)

→integer(4 bytes)

→float(8 bytes)

31. Find the output(datatypes)

```
void main()
{
   char ch=291;
   printf("%d %d %c\n",2147483648,ch,ch);
   return 0;
}
```
Output : -2147483648 35 #

Explanation :

→Here we get -ve output due to integer overflow(i.e exceeding max.size).

→Next we get 35 due to character overflow(-128 - 127(total 256 combinations can be represented in char)) so the compiler prints 35 by taking modulo between 291%256 = 35

→now we get # because the ascii value of # is 35 because we are printing character(%c)

32. Find the output(datatypes)

```
void main()
{
   int g;
   g=300000*300000/300000;
   printf("g=%d\n",g);
}
```
Output : -647

Explanation :

→here the Output is -647 because of floating point arithmetic.

→ The execution occurs from left to right(as * and / as same precedence)

→the product of 300000 * 300000 exceeds the range so we get an -ve number(to check convert the product in binary and check the MSB ==1,get 2's complement of the number) and division is done with that number we get -647

→Answer can be compiler specific for number bits for integer storage.

33. Find the output(datatypes)

```
void main()
{
   float a;
a=4/2;
printf("%f %f\n",a,4/2);

}
```

Output : 2.000000 0.000000

Explanation :

→as a is float a = (4/2 → 2.000000) but,

→in the print statement we are printing 4/2 where 4 and 2 is integer so the answer is 2 but we are printing it as float(i.e using %f) so output is 0.000000.

34. Find the output(operator and expression)

```
void main(){
        printf("%d\n",sizeof(4)/sizeof(2.0));
        printf("%d\n",sizeof(2.0)/sizeof(4));
}
```

Output : 0 2

Explanation :

→here sizeof is an unary operator which has high precedence than division(/) so the order of execution is sizeof(4) = 4(depends on compiler) and sizeof(2.0) = 8

→4/8 == 0(as we are printing integer i.e , %d used)(1st print statement)

→8/4 == 2(as we are printing integer i.e , %d used)(2nd print statement)

35. Find the output(operator and expression)

```
void main()
{
    int x=10,y=5,p,q;
    p=x > 9;
    q=x>3&& y!=3;
    printf("p=%d q=%d \n",p,q);

}
```

Output : p = 1 q=1

Explanation :

→ first x > 9 is true i.e p = 1 as x is an non-negative integer.

→ in q = x>3 && y != 3, first relational operations are computed as they have high precedence so the execution order is x>3(1), y != 3(1) then q = 1 && 1 which is 1

→ therefore p = 1 & q= 1.