TRICKS IN C PROGRAMMING

2)CONDITION AND LOOPS TRICKS

1)
```
int main(void)
{
  int i=0;
  for(i=0; i<5; i++)
  {
      int i=5;
      printf("%d ",i);
  }
     return 0;
}
```

```
Output:5 5 5 5 5.
Explanation:at each iteration i
change to 5 before printing.
```

………

2)
```
int main(void)
{
 int i=10;
 static int j=i;
 if(j==i)
    printf("equal");
 else if(j>i)
    printf("j is greater");
 printf("i is greater");
      return 0; }
```

```
Output:compiler error.
Explanation:static variables must
initialize during declaration with
constatnt value.
Example:
Static int x=10;
```

………

3)
```
 int main()
{
    int x;
    for(x=-1; x<=10; x++)
    {
        if(x < 5)
            continue;
        else
            break;
        printf("IndiaBIX");
    }
    return 0;
}
```

```
Output:No output.
Explanation:at first iteration x=-1.
X<=10(true).if(x<5)->true then
continue(mean jump to x++). This
process will continue till x=4. When
x become 5 if condition will fail and
jump to else. When come to else we
found break statement(mean exit from
loop). So no output will print on the
terminal.
```

………

4)Which of the following cannot be checked in a `switch-case` statement?

a)character.
b)integer.
c)float.
4)enum.

Output:float.
Explanation:The `switch/case` statement in the c language is defined by the language specification to use an `int` value, so you can't use a `float` value.

```
switch( expression )
{
    case constant-expression1:    statements 1;
    case constant-expression2:    statements 2;
    default : statements 4;}
```

5)
```
int main()
{
    int i=0;
    for(; i<=5; i++);
      printf("%d", i);
    return 0;
}
```

Output:6.

Explanation: **Step 1**: `int i = 0;` here variable `i` is an integer type and initialized to '0'.

Step 2: `for(; i<=5; i++);` variable `i=0` is already assigned in previous step. The semi-colon at the end of this `for` loop tells, "there is no more statement is inside the loop".

Loop 1: here `i=0`, the condition in for(; 0<=5; i++) loop satisfies and then `i` is incremented by '1'(one)

Loop 2: here `i=1`, the condition in for(; 1<=5; i++) loop satisfies and then `i` is incremented by '1'(one)

Loop 3: here `i=2`, the condition in for(; 2<=5; i++) loop satisfies and then `i` is incremented by '1'(one)

Loop 4: here `i=3`, the condition in for(; 3<=5; i++) loop satisfies and then `i` is increemented by '1'(one)

Loop 5: here `i=4`, the condition in for(; 4<=5; i++) loop satisfies and then `i` is incremented by '1'(one)

Loop 6: here `i=5`, the condition in for(; 5<=5; i++) loop satisfies and then `i` is incremented by '1'(one)

Loop 7: here `i=6`, the condition in for(; 6<=5; i++) loop fails and then `i` is not incremented.

Step 3: `printf("%d", i);` here the value of `i` is 6. Hence the output is '6'.

....

6)
```
int main()
{
    char str[]="C-program";
    int a = 5;
  printf(a >10?"Ps\n":"%s\n", str);
    return 0;
}
```

Output: C-program.
Explanation:
Step 1: `char str[]="C-program";` here variable `str` contains "C-program".
Step 2: `int a = 5;` here variable `a` contains "5".
Step 3: `printf(a >10?"Ps\n":"%s\n", str);` this statement can be written as:
```
if(a > 10)
{
    printf("Ps\n");
}
else
{
    printf("%s\n", str);
}
```

....

7)
```
int main()
{
  int a = 500, b = 100, c;
    if(!a >= 400)
        b = 300;
    c = 200;
    printf("b = %d c = %d\n", b, c);
    return 0; }
```

```
Output:b=100 c=200.
Explanation:first step if(!a>=400)->if(!500>=400)
->if(0>=400)->false so b doesn't change and still 100
```

8)
```
int main()
{
   int n;
   for(n = 7; n!=0; n--)
     printf("n = %d", n--);
   getchar();
   return 0;
}
```

Output:infinite loop.
Explanation: because n is never zero when loop condition (n != 0) is checked.this is because post decrement in printf function.

....

9)
```
int main()
{
   int i = 1;
   do
   {
      printf("%d\n", i);
      i++;
      if (i < 15)
        continue;
   } while (0);
   return 0;
}
```

Output:1.
Explanation: The do wile loop checks condition after each iteration. So after continue statement, control transfers to the statement while(false). Since the condition is false 'i' is printed only once.

....

10)
```
int main()
{
   int i = 1;
   do
   {
     printf("%d\n", i);
     i++;
     if (i < 15)
       break;
     } while (1);
     return 0;
}
```

Output:1.
Explanation:when enters the loop will print the value of i(1) then increment I so I become 2 and check for i<15 true then break the loop and exit.

....

11)
```
/* Assume 2 byte integer*/
int main()
{
    unsigned int i = 65535;
    while(i++ != 0)
        printf("%d",++i);
    printf("\n");
    return 0;
}
```

Output:infinite loop.
Explanation: Here `unsigned int` size is 2 bytes. It varies from 0,1,2,3, ... to 65535. We will check condition in loop we found it 'i' is post increment so store value then increment so 65536!=0->true and then increment 'i'(65536) overflow will ocuur and 'i' will become 0 then increment 'i'(pre increment) in printf then 'i' become 1 then print value of i
And we back to condition in loop, now i=1, 1!=0 true and so on i never become 0.

12)
```
int main()
{
   int i=0;
   for(i=0; i<20; i++)
   {
     switch(i)
     {
       case 0:
         i+=5;
       case 1:
         i+=2;
       case 5:
         i+=5;
       default:
         i+=4;
         break;
     }
     printf("%d  ", i);
   }
   }
```

Output:16 21.
Explanation: Initially i = 0. Since case 0 is true i becomes 5, and since there is no break statement till last statement of switch block, i becomes 16. Before starting the next iteration, i becomes 17 due to i++. Now in next iteration no case is true, so execution goes to default and i becomes 21.

In C, if one case is true switch block is executed until it finds break statement. If no break statement is present all cases are executed after the true case. If you want to know why switch is implemented like this, well this implementation is useful for situations like below.

```
 switch (c)
{
   case 'a':
   case 'e':
   case 'i' :
   case 'o':
   case 'u':
     printf(" Vowel character");
     break;
   default :
     printf("Not a Vowel character");; break;
 }
```

……

13)
```
 int main()
{
     int i;
     i = 1, 2, 3;
  printf("i = %d\n", i);
     return 0;
}
```

Output: 1
Explanation: The above program prints 1. Associativity of comma operator is from left to right, but = operator has higher precedence than comma operator.
Therefore the statement i = 1, 2, 3 is treated as (i = 1), 2, 3 by the compiler.

……

14)
```
int main()
{
    int x = 3;
    float y = 3.0;
    if(x == y)
        printf("x and y are equal");
    else
        printf("x and y are not equal");
    return 0;
}
```

Output: x and y are equal.
Explanation:when compare int with float int convert to float so they will be equal.

15)
```
int main(void)
{
  int i=5;
  switch(i)
  {
      case '5':printf("hello");
      break;
      case '10':printf("world");
      break;
      default:printf("hey default");
  }
     return 0;
}
```

Output: hey default.
Explanation:I is integer and cases is characters so 5!='5'. if we want to print hello, we must do this char i='5'.

………

16)
```
int main()
{
    short int i = 0;
    for(i<=5 && i>=-1; ++i; i>0)
        printf("%u,", i);
    return 0;
}
```

++i is the condtion of loop
Output:1……32767 then overflow will ocur and i become -32768 and increment till i=-1 then ++I will be 0 and exit the loop.

………

17)
```
int main()
{
   while(1)
{

if(printf("%d",printf("mohamed")))
       break;
   else
        continue;
  }
    return 0;
}
```

Output: mohamed7
Explanation:first printf will print Mohamed and then return number of characters in Mohamed which is 7 characters so if condition will be true and then break will exit the loop.

………

18)
```
int main()
{
    unsigned int i=10;
    while(i-- >= 0)
        printf("%u ",i);
    return 0;
}
```

Output:infinite loop
Explanation:because of unsigned the value of i is never negative

```
 19)
int main()
{
    char ch;
    if(ch = printf(""))
        printf("It matters\n");
    else
        printf("It doesn't matters\n");
    return 0;
}
```

Output: It doesn't matters.
Explanation: we know that printf returns number of characters so printf in the if condition reurns 0 so if condition will fail and else statement will run.

……

```
20)
 int main()
{
/* Assume 2 byte integer*/
    unsigned int i = 65536;
    while(i != 0)
        printf("%d",++i);
    printf("\n");
    return 0;
}
```

Output:NO Output.
Explanation:i=65536 larger than max value of unsigned int so overflow will occur and I will become 0 so while condtion will fail.

……

```
21)
int main()
{
    int x,y=2,z;
    if ( x = y%2)
         z =2;
    printf("%d %d ",z,x);
    return 0;
}
```

Output: z(garbage value), x(0)
Explanation: y%2 is zero and it'll be assigned to x. So the value of x becomes zero which is also the effective condition for if. And therefore, condition of if is false.

……

```
22)
int main()
{
    int i=4;
    switch(i&1)
    {
 case 0:
    printf("even");
    break;
 case 1:
    printf("odd");
    break;
    }
    return 0;
}
```

Output:even
Explanation:4&1->00000100&00000001=0 so will print even.
i&1 returns 1 for odd values.if we put any even value to I the output will be even otherwise odd.

……

23)
```
int main()
{
    int num=5;
    if(num=3)
        printf("num=%d",num);
    else
        printf("num=%d",num);
    return 0;
}
```

Output:num=3.
Explanation:notice that operator inside if is assignment operator not equality operator so num become 3 and if condition will be true

………

24)
```
int y=12;
int main()
{
    int x=2;
    while(x<9)
    {
       y=13;
       y++;
       x++;
    }
        printf("%d",y);
    return 0;
}
```

Output:14
Explanation: first initialize y=12 and x=2 condition in the loop is true as 2<9. In the body of while we assign 13 to y so y now is 13 then increment y so now y is 14 and increment x so x is now 3.return to condition 3<9 true. In the body of while we assign 13 to y so y now is 13 and increment it so y now is 14 and increment x so x is now 4 and so on till x=9.

………

25)
```
const int y=12;
int main()
{
    int x=2;
    while(x<9)
    {
       y++;    //error
       x++;
    }
        printf("%d",y);
    return 0;
}
```

Output:compiler error.
Explanation: y is constant so we can't change.so we can't increment it's value.

………

26)
```
int main()
{
    int x=1, y=1;
    for(; y; printf("%d %d\n", x, y))
    {
        y = x++ <= 5;
    }
    printf("\n");
    return 0;
}
```

Output:
```
2 1
3 1
4 1
5 1
6 1
7 0
```

```
27)
#define L 2
int main()
{
    auto int a=2;
    switch(a,a*5)
    {
    case L:
        printf("ABC\n");
        break;
    case L*2:
        printf("EFG\n");
        break;
    case L*3:
        printf("HIJ\n");
        break;
    default:
        printf("KLM\n");
    case L*4:
        printf("NOP\n");
        break;
    }
    return 0;
}
```

```
Output:
KLM
NOP
```
Explanation:inside switch comma is separator so a*5 will be in case switch(10). There is not case 10 so default will run and there is no break after default so the case after default will run.

……

```
28)
int main()
{
   char c1='a',c2='A';
   int i=c2-c1;
   printf("%d",i);
    return 0;
}
```

Output:-32
Explanation:Ascii value of a is 97 and ascii for A is 65 so 65-97=-32

……

```
29)
int main()
{
   unsigned int num;
   int i;
   scanf("%u",&num);
   for(i=0; i<16; i++)
    printf("%d",(num<<i & 1<<15)?1:0);
    return 0;
}
```

Output:it prints binary equivalent num.

……

```
30)
int main()
 {
   char i=0;
   for(i=0; i<1000; i++)
    printf("hello");
    return 0; }
```

Output:infinite loop.
Explanation:max value of i is 127, so condition will always true.

31)
```
int main()
{
   short a=30000,b=40000;
   int c=a+b;
   printf("%d",c);
    return 0;
}
```

Output:4464.
Explanation:range of short from -32768 to 32767 so in b overflow will occur so b=40000-65536(all range)=-25536.
C=30000-25536=4464.

……

32)
```
int main()
{
   char c=48; //ascii value is
0
   int i,mask=01;
   for(i=1; i<=5; i++)
   {
     printf("%c", c|mask);
     mask <<=1;
   }
    return 0;
}
```

Output:12480
Explanation:at i=1 -> i<=5(true)
c|mask->0011000(48)|00000001=49(in deciaml) here we want to print %c so 49 will be 1 and mask<<1==000000010.
At i=2-> 2<=5(true)
c|mask->00110000|00000010=00110010=50(in decimal). here we want to print %c so 50 will be 2 and
mask<<1==000000100.and so on

……

33)
```
int main()
{
    int i = 5;
    while(i-- >= 0)
        printf("%d,", i);
    i = 5;
    printf("\n");
    while(i-- >= 0)
        printf("%i,", i);
    while(i-- >= 0)
        printf("%d,", i);
    return 0;
}
```

Output:
4,3,2,1,0,-1
4,3,2,1,0,-1.
Explanation:First i=5 i–(post decrement)>=0(true)
I will decrement so print 4 and so on till i=-1, so print 4,3,2,1,0,-1.
After the first while we reassign i with 5 and repeat the above process so print 4,3,2,1,0,-1
Now i=-1; in the last while -1>=0 false.

……

34)
```
int main()
{
  unsigned int m=33;
  printf("%x", ~m);
   return 0;
}
```

Output:ffffffde
Explanation:
m=33->000000000000000000000000 00100001
~m->11111111 11111111 11111111 1101 1110=ffffffde

35)which is faster the first or the second?

```
int main()
{
  int c1 = 0, c2 = 0;
  /* FIRST */
  for(int i=0;i<10;i++,c1++)
    for(int j=0;j<100;j++, c1++);
      //do something
  /* SECOND */
  for(int i=0; i<100; i++, c2++)
      for(int j=0; j<10; j++, c2++);
        //do something
  printf("Count in FIRST =%d\n",c1);
  printf("Count in FIRST =%d",c2);
  return 0;
}
```

Count in FIRST =1010.
Count in FIRST =1100.
So the first one is faster.

……

36)

```
int main()
{
    int i=3;
    switch(i)
    {
        case 1:
            printf("Hello\n");
        case 2:
            printf("Hi\n");
        case 3:
            continue;
        default:
            printf("Bye\n");
    }
    return 0;
}
```

Output:Compiler error.
Explanation:we can't use continue in switch.

……

37)

```
int main()

{
    int x = 10, y = 20;
    if(!(!x) && x)
        printf("x = %d\n", x);
    else
        printf("y = %d\n", y);
    return 0;
}
```

Output:10
Explanation:x=10->!x=0->!0=1 so if condition will be true. So print x=10.

```
 38)
int main()
{
    int i=4;
    switch(i)
    {
        default:
           printf("This is default\n");
        case 1:
           printf("This is case 1\n");
           break;
        case 2:
           printf("This is case 2\n");
           break;
        case 3:
           printf("This is case 3\n");
    }
    return 0;
}
```

```
Output:
This is default
This is case 1
Explanation: there is not
break after default.
```

………

```
39)
int main()
{
    int i = 1;
    switch(i)
    {
        printf("Hello\n");
        case 1:
            printf("Hi\n");
            break;
        case 2:
            printf("\nBye\n");
            break;
    }
    return 0;
}
```

```
Output:Hi
```
Explanation: switch(i) has the variable i it has the value '1'(one).

Then case 1: statements got executed. so, it prints "Hi". The break; statement make the program to be exited from switch-case statement.

switch-case do not execute any statements outside these blocks case and default
Hence the output is "Hi"

………

```
40)
int main()
{
    int i=1;
    for(;;)
    {
        printf("%d ", i++);
        if(i>10)
           break;
    }
    return 0; }
```

```
Output:1 2 3 4 5 6 7 8 9 10
Explanation:we will print I till i=11.
```

41)
```
int main()
{
    int movie = 1;
    switch (movie << (2 + movie))
    {
    default:
        printf(" Traffic");
    case 4:
        printf(" Sultan");
    case 5:
        printf(" Dangal");
    case 8:
        printf(" Bahubali");
    }
}
```

Output: Bahubali
Explanation:1<<(2+1)=1<<3=8

………

42)
```
int main()
{
    switch(2)
    {
    case 1L:
        printf("No");

    case 2L:
        printf("%s","GEEKS");
        goto Love;

    case 3L:
        printf("Please");

    case 4L:Love:
        printf("FOR");
    }
}
```

Output :GEEKSFOR
Explanation: It is possible to write label of goto statement in the case of switch case statement.

………

43)
```
int main()
{
    int i = 0, j = 0;
    while (i<5,j<10)
    {
        i++;
        j++;
    }
    printf("%d %d", i, j);}
```

Output: 10 10.
Explanation :Here, both the expression before and after "," operator will be evaluated but the expression right will be returned, i.e. the loop will be ended if the condition,

j < 10becomes false.

44)
```c
int main()
{
    int a = 10;
    switch(a)
    {
    }
    printf("This is c program.");
    return 0;
}
```

Output: This is c program.
Explanation:we can use switch case without cases.

……

45)
```c
int  main()
{
    int  i = 0, j = 0;
    while  (i<5 & j<10)
    {
        i++;
        j++;
    }
    printf("%d %d", i, j);
}
```

Output:5 5.
Explanation: The loop will execute only if both the conditions will be true

……

46)
```c
int main()
 {
    int i = 0, j = 0;
    for (i = 0; i < 5; i++)
    {
        for (j = 0; j < 1;)
        {
            break;
        }
        printf("GeeksQuiz \n");
    }
}
```

Output: GeeksQuiz will print 5 times.
Explanation: Due to break, the inner loop doesn't execute

……

47)
```c
int main()
{
    int i=1;
    while()
    {
        printf("%d\n", i++);
        if(i>10)
           break;
    }
    return 0;
}
```

Output:compiler error.
Explanation:there is not condition in while loop.

```
48)
int main()
{
    int a = 5;
    switch(a)
    {
    case 1:
    printf("First");
    case 2:
    printf("Second");
    case 3 + 2:
    printf("Third");
    case 5:
    printf("Final");
    break;
    }
    return 0;
}
```

Output:compiler error.
Explanation:duplicate case value

………

```
49)
int main()
{
    int P = 10;
    switch(P)
    {
       case p: //error must be constant
       printf("Case 1");

       case 20:
       printf("Case 2");
       break;

       case 20:
       printf("Case 3");
       break;
    }
    return 0;
}
```

Output:compiler error.
Explanation:case must be constant.

………

```
50)
int main()
{
    int a = 10, b;
    a >=5 ? b=100: b=200;
    printf("%d\n", b);
    return 0;
```

Output:compiler error.
Explanation:Lvalue required must be (b=100):(b=200)

```
}
```

51)

```
int main()
{
    int i = 10, j = 20;
    if(i = 5) && if(j = 10)
        printf("Have a nice day");
    return 0;
}
```

```
Output:compiler error.
Explanation:must be like this
If(i=5 && j=10)
```

……

52)

```
int main()
{
    int i = 10, j = 15;
    if(i % 2 = j % 3)
        printf("IndiaBIX\n");
    return 0;
}
```

```
Output:compiler error.
Explanation:i%2=0 and j%3=0 we can't
assign value to value
```

……

53)

Write 2 codes for infinite loop?

```
1)for(; ;)
2)while(1)
```

……

54)how to swap two numbers by using 3 methods?

```
int main()
{
   int x=10,y=20,temp;
   temp=x;
   x=y;
   y=temp;
printf("x=%d,y=%d",x,y);
    return 0;
}
```

```
int main()
{
   int x=10,y=20;
   x=x+y;  //x=30
   y=x-y;  //y=30-20=10
   x=x-y;  //x=30-10=20
  printf("x=%d,y=%d",x,y);
    return 0;
}
```

```
int main()
{
   int x=10,y=20;
   x=x^y;
   y=x^y;
   x=x^y;
  printf("x=%d,y=%d",x,y);
    return 0;
}
```

……

54)

73)

```
int main()
 {
   int x=0;
   if(x++)
    printf("true");
   else
    printf("false");
    return 0;
}
```

```
Output:false.
Explanation: x++ is post
increment so stire value
then increment so if(0) will
fail and else will run so
print fals. If we want to
print true we must use
if(++x).
```

```
55)
int main()
{
   int x=97;
   switch(x)
   {
   case 'a':
    printf("yes");
    break;
   case 97:
    printf("no");
    break;
   }
    return 0;
}
```

Output:compiler error.
Explanation: duplicate case value as ascii of 'a' is 97.

………

```
56)
#define max(a) a
int main()
 {
   int x=1;
   switch(x)
   {
   case max(2):
    printf("yes");
   case max(1):
    printf("no");
    break;
   }
    return 0;
}
```

Output:no
Explanation:this is macro(text replacment) so max(2) will replace by 2 and max(1) will replace by 1 so will print no.

………

```
57)
int main()
{
 unsigned char i;
 for(i=5; i>=0; i--)
    printf("%d ",i);
    return 0;
}
```

Output:infinite loop.
Explanation: 'i' is unsigned char so it's value can't accept negative values so condition always true.

………

```
58)
int main()
{
    int a=2;
    if(a--,--a,a)
        printf("hello");
    else
        printf("hi");
    return 0;
}
```

Output:hi.
Explanation:a—(post decremenr) so store 2 and then a will decrement to 1 after that –a(pre decrement) so decrement a to 0 then store value and finally a is 0 so if condition will fail.

```
59)
int main()
{
  int x=1;
  if(x--)
    printf("hello guys");
    --x;             //error here
    else
        printf("%d",x);

    return 0;
}
```

Output:compiler error
Explanation:else without if as –x is statement and else must be after if directly.

………

```
60)
int main()
{
  int x=3;
  if(x==2); //notice semicolon
  x=0;
  if(x==3)
    x++;
  else
    x+=2;
  printf("%d",x);
    return 0; }
```

Output:2
Explanation: we define x with 3, we check for if(x==2) then we assign zero to x so now x is 0, we check for x==3 false so else will execute x+=2, and x is 0 so x=x+2=0+2=2.

………

```
61)
int main()
{
 int i=3;
 while(i--) //notice i–not i
 {
     int i=100;
     i--;
     printf("%d ",i);
 }
    return 0; }
```

Output:99 99 99.
Explanation: we define i=3 and check while(i--)->true and inside while define another variable i=100 and decrement it by 1 so we print 99 and repeat this process three times so print 99 3 times.

………

```
62)
int main()
{
    int i;
    if(printf("0"))
        i=3;
    else
        i=5;
    printf("%d",i);
    return 0; }
```

Output:03.
Explanation: printf will print 0 and return number of characters so if condition is true so i=3 will be execute so output 03.

63)
```
int main()
{
    int c=5,no=10;
    do{
        no/=c;
    }while(c--);
    printf("%d",no);
    return 0; }
```

```
Output: runtime error
Explanation: in the last iteration
will divide in zero.
```

……

64)
```
int x;
int main()
{
   if(x)
    printf("gemy");
   else
    printf("hi");
    return 0; }
```

```
Output: hi
Explanation: when we define global
variable without initialize it the
compiler initialize it with zero. So
if(0) is false and hi will print.
```

……

65)
```
int main()
{
  int a=5;
  switch(a)
  {
  default:
    a=4;
case 6:
    a--;
case 5:
    a=a+1;
case 1:
    a--;
  }
  printf("%d",a);
    return 0; }
```

```
Output: 5
Explanation: first we will increment
a, so a become 6 and there is not
break after case 5 so case 1 will be
executed so a will decrement by 1 so
a become 5.
```

……

66)what is the function of below code?
```
int main()
{ int num,prod=1;
   scanf("%d",&num);
   while(num)
   {
    prod*=num;
    num--;
   }
   printf("%d",prod);
   return 0; }
```

```
Output: factorial of num
Factorial(4)=4*3*2*1=24.
Factorial(5)=5*4*3*2*1=120
```

66)

```
int main()
{
    int i,j,rows=4;
    for(i=rows; i>=1; i--)
    {
        for(j=1; j<=i; j++)
            printf("%d ",j);
        printf("\n");
    }
    return 0;
}
```

```
Output:
1 2 3 4
1 2 3
1 2
1
```

……

67)

```
int main()
{
    int i = 1024;
    for (; i; i >>= 1)
        printf("hi\n");
    return 0;
}
```

```
Output:print hi 11 times.
Explanation:i=1024 ->print hi
i>>=1 = 1024/2=512.
i=512 ->print hi.
i>>==512/2=256.
i=256->print hi and so on till i=0;
```

……

68)

```
int main()
{
   int n;
   for (n = 9; n!=0; n--)
     printf("n = %d", n--);
   return 0;
}
```

```
Output:infinite loop.
Explanation:because of n-- in printf.
```

……

69)

```
int main()
{
  char check='a';
  switch(check)
  {
  case 'a' || 1:
    printf("case1");
  case 'b' || 1:
    printf("case2");
  default:
    printf("Default");
  }
   return 0;
}
```

```
Output:compiler error.
Explanation:duplicate case value why?
'a' || 1 return 1.
'b' || 1 return 1.
```