AnswerKey

1. **Error : L-value required**

`++/--` operator works on variables only.

2. **Error : increment of read-only variable 'var'.**

`++/--` operator works on variables only, we can not change the value of a `const`.

3. **45**
1) expand the expression : `x=x+(x++)+(++x)+x;`
2) due to pre increment `++x` , `x` will be 11 for this expression.
3) after executing expression `x` will be 44.
4) finally `x` will be 45 due to post increment (`x++`).
Note: the output of pre and post increment based operators may not same on all the compilers, in GCC Linux compiler output will be 46 and in TurboC output will be 45.

4.
value is =80

5.
var : E , 69
```
unsigned short var='B';
var has ASCII value of 'B' which is 66
var+=2 results to var=68
var++ results to var =69 which is ASCII value of 'E'
thus its prints E when placeholder is %c & 69 when placeholder is %d
```

6.
Error : L-Value required.

7.
x=1
1)`(20 || 40)` .... both are non zero values, will return 1.
2)`(1) && 10` .... both are non zero values, hence output will be 1.

8.
AA1
AABB1
1) `printf()` returns total number of characters, `printf("AA")` will return 2 hence expression is true second expression `printf("BB")` will not execute.
2) In this statement both expressions will execute.

9.
1,2

Since the associability is left to right for `==` operator, thus `(a==b)` is evaluated first which results in 0 as `a =3 & b=2`
Now 0==0 which assigns 1 to a
Thus finally a=1 & b=2

10.
Value of intVar=23, x=21
Since assignment operator (`=`) has more precedence than comma operator, so `=` operator will be evaluated first.
Here, `x = ++intVar`, `intVar++`, `++intVar`;
`++intVar` will be assigned to `x` then comma operator will be evaluated.

11.
-6

```
250 is beyond the range of  char, the value of val will be -6.
Consider the expression:
ans= val+ !val + ~val + ++val;
Operator! , ~ and ++ have equal precedence. And it associative is
right to left.
So, First ++ operator will perform the operation. So value val will -5
Now,
ans = -5 + !-5 + ~-5 + -5
= -5 + !-5 + 4 - 5
= -5 + 0 + 4 -5
= -6
```

12.
Error: L-value required.
`(int)a` will return an integer constant value and 10 is also an integer value, and constant value can not assign in constant value.

13.
x=100
x=50
Since `=` (assignment operator) has more precedence than comma operator (`,`), so `=` operator evaluates first and 100 will be assigned to `x`.
In second case, `x=(100,30,50)`, here `()` have more precedency so `(100,30,50)` will be evaluated first from left to right, and `x` will be 50.

14.
0 0 1 3 1

```
inti=-1,j=-1,k=0,l=2,m;
m=i++&&j++&&k++||l++;

++ Operator has precedence over &&, || operator,
thus it is evaluated first. So the expression turns to be
m= 0 && 0 && 1 || 3
```

```
i=0. j=0, k=1, l=3
Now && has precedence over || and both has associativity
left to right. Hence the expression part 0 && 0 is evaluated first, so
on.
m=((0 && 0) && 1) || 3)
 =((0 && 1) || 3)
= 1 || 3= 1
Thus value of
m=1
i=0
j=0
k=1
l=3
```

15.

value of var= 10

value of var= 10

```
Here unary minus (or negation) operator is used twice.
According to the math rules:
- -10 = 10
+ +10=10
```

16.

x=200,y=100

In expression `x=(100,200);` 200 will be assigned to `x` because `()` has precedence over `=` and `,` operators. Hence `()` will be evaluated first and value 200 is assigned to `x` since `','` has left to right associativity. On the other hand, for expression `y=100,200;` here priority of `=` is higher than `','` operator so 100 will be assigned to `y`.

17.

->, %, +, =

Link: https://www.includehelp.com/c/operators-aptitude-questions-and-answers.aspx