```c
#include<stdio.h>
int main(){
    int a=2,b=7,c=10;
    c=a==b;
    printf("%d",c);
    return 0;
}
```

equal "==" operator checks if the operands are equal

c = False = 0

(A) 0
(B) 1
(C) 7
(D) 2
(E) Compilation error

Answer :(A)

Explanation:
== is relational operator which returns only two values.
0: If a == b is false
1: If a == b is true
Since
a=2
b=7
So, a == b is false hence b=0

7.
What will be output of the following c program?

```c
#include<stdio.h>
int main(){
    int max-val=100;
    int min-val=10;
    int avg-val;
    avg-val = max-val + min-val / 2;
    printf("%d",avg-val);
    return 0;
}
```

5

105

but, the variable names have "-"(hyphens) invalid!

(A) 55

(B) 105
(C) 60
(D) Compilation error
(E) None of these

Answer: (D)
Explanation:
We cannot use special character - in the variable name.

8.
What will be output when you will execute following c code?

```
#define PRINT printf("Star Wars");printf(" Psycho");
#include<stdio.h>
int main(){
    int x=1;
    if(x--)
         PRINT
    else
         printf("The Shawshank Redemption");
    return 0;
}
```

cosists of 2 statements
∴ breaks the if-else ladder

Choose all that apply:
(A) Stars Wars Psycho
(B) The Shawshank Redemption
(C) Warning: Condition is always true
(D) Warning: Condition is always false
(E) Compilation error

Answer: (E)

Explanation:
PRINT is macro constant. Macro PRINT will be replaced by its defined statement just before the actual compilation starts. Above code is converted as:

```
int main(){
    int x=1;
```

```
    if(x--)
         printf("Star Wars");
printf(" Psycho");
    else
         printf("The Shawshank Redemption");
}
```

If you are not using opening and closing curly bracket in if clause, then you can write only one statement in the if clause. So compiler will think:

(i)

```
if(x--)
    printf("Star Wars");
```

It is if statement without any else. It is ok.

(ii)

```
printf(" Psycho");
```

It is a function call. It is also ok

(iii)

```
else
         printf("The Shawshank Redemption");
```

You cannot write else clause without any if clause. It is cause of compilation error. Hence compiler will show an error message: Misplaced else

Explanation:
As we know in c zero represents false and any non-zero number represents true. So in the above code:

(0.001 - 0.1f) is not zero so it represents true. So only if clause will execute and it will print: David Beckham on console.
But it is bad programming practice to write constant as a condition in if clause. Hence compiler will show a warning message: Condition is always true

Since condition is always true, so else clause will never execute. Program control cannot reach at else part. So compiler will show another warning message: Unreachable code

?! 9.
What is meaning of following declaration?
int(*ptr[5])();

Osborne book page 129 (pdf pg: 164)

function name

return type

(A) ptr is pointer to function.
(B) ptr is array of pointer to function.
(C) ptr is pointer to such function which return type is array.
(D) ptr is pointer to array of function.
(E) None of these

Answer: (B)
Explanation:
Here ptr is array not pointer.

?! 10.
What will be output when you will execute following c code?

```
#include<stdio.h>
int main(){
     int const X=0;
     switch(5/4/3){
        case X:  printf("Clinton");
                  break;
        case X+1:printf("Gandhi");
                  break;
        case X+2:printf("Gates");
                  break;
        default: printf("Brown");
     }
     return 0;
}
```

Choose all that apply:

(A) Clinton
(B) Gandhi
(C) Gates
(D) Brown
(E) Compilation error

Answer: (E)

Explanation:
Case expression cannot be constant variable.

?! 11.
What will be output when you will execute following c code?

Osborne book page 198 (pdf pg: 233)

```
#include<stdio.h>
enum actor{
    SeanPenn=5,
    AlPacino=-2,
    GaryOldman,  <- -2+1=-1   initially
    EdNorton     <- -1+1=0
};
int main(){
     enum actor a=0;
     switch(a){
        case SeanPenn:  printf("Kevin Spacey");
                         break;
        case AlPacino:  printf("Paul Giamatti");
                         break;
        case GaryOldman:printf("Donald Shuterland");
                         break;
        case EdNorton:  printf("Johnny Depp");
     }
     return 0;
}
```

Choose all that apply:

(A) Kevin Spacey

(B) Paul Giamatti
(C) Donald Shuterland
(D) Johnny Depp
(E) Compilation error

Answer: (D)

Explanation:
Default value of enum constant
GaryOldman = -2 +1 = -1
And default value of enum constant
EdNorton = -1 + 1 = 0
Note: Case expression can be enum constant.

12. What will be output of following c code?

```
#include<stdio.h>
extern int x;
int main(){
    do{
        do{
            printf("%o",x);
        }
        while(!-2);
    }
    while(0);
    return 0;
}
int x=8;
```

Thus executes once

False

False

→ in octal (%o) = 10

(A) 8
(B) 10
(C) 0
(D) 9
(E) Compilation error

Answer: (B)

Explanation:

Here variable x is extern type. So it will search the definition of variable x. which is present at the end of the code. So value of variable x =8
There are two do-while loops in the above code. AS we know do-while executes at least one time even that condition is false. So program control will reach at printf statement at it will print octal number 10 which is equal to decimal number 8.
Note: %o is used to print the number in octal format.
In inner do- while loop while condition is! -2 = 0
In C zero means false. Hence program control will come out of the inner do-while loop. In outer do-while loop while condition is 0. That is again false. So program control will also come out of the outer do-while loop.

13.
What will be output of following c code?

```
#include<stdio.h>
int main(){
    static int i;
    for(++i;++i;++i) {
         printf("%d ",i);
         if(i==4) break;
    }
    return 0;
}
```

space

(A) 4
(B) 24
(C) 25
(D) Infinite loop
(E) Compilation error

Answer: (b)

Explanation:

Default value of static int variable in c is zero. So, initial value of variable i = 0
First iteration:
For loop starts value: ++i i.e. i = 0 + 1 = 1
For loop condition: ++i i.e. i = 1 + 1 = 2 i.e. loop condition is true. Hence printf statement will print 2
Loop incrimination: ++I i.e. i = 2 + 1 =3
Second iteration:
For loop condition: ++i i.e. i = 3 + 1 = 4 i.e. loop condition is true. Hence printf statement will print 4. Since is equal to four so if condition is also true. But due to break keyword program control will come out of the for loop.

14.
What will be output of following program?

```
#include<stdio.h>
#include<string.h>
int main(){
    char *ptr1 = NULL;
    char *ptr2 = 0;
    strcpy(ptr1," c");
    strcpy(ptr2,"questions");
    printf("\n%s %s",ptr1,ptr2);
    return 0;
}
```

(A) c questions
(B) c (null)
(C) (null) (null)
(D) Compilation error
(E) None of above

Answer: (C)

Explanation:
We cannot assign any string constant in null pointer by strcpy function.

15.
What will be output of following c code?

```c
#include<stdio.h>
int main(){
    int *p1,**p2;
    double *q1,**q2;
    printf("%d %d ",sizeof(p1),sizeof(p2));
    printf("%d %d",sizeof(q1),sizeof(q2));
    getch();
    return 0;
}
```

(A) 1 2 4 8
(B) 2 4 4 8
(C) 2 4 2 4
(D) 2 2 2 2
(E) 2 2 4 4

Answer: (D)

cause, any poinyer will just be holding the address

Explanation:
Size of any type of pointer is 2 byte (In case of near pointer)

16.
What will be output if you will compile and execute the following c code?

```c
#include<stdio.h>
int main(){
    char huge *p=(char *)0XC0563331;
    char huge *q=(char *)0XC2551341;
    if(p==q)
        printf("Equal");
    else if(p>q)
        printf("Greater than");
    else
        printf("Less than");
```

```
    return 0;
}
```

(A) Equal
(B) Greater than
(C) Less than
(D) Compiler error
(E) None of above

Answer: (A)

Explanation:
As we know huge pointers compare its physical address.
Physical address of huge pointer p
Huge address: 0XC0563331
Offset address: 0x3331
Segment address: 0XC056
Physical address= Segment address * 0X10 + Offset address
=0XC056 * 0X10 +0X3331
=0XC0560 + 0X3331
=0XC3891
Physical address of huge pointer q
Huge address: 0XC2551341
Offset address: 0x1341
Segment address: 0XC255
Physical address= Segment address * 0X10 + Offset address
=0XC255 * 0X10 +0X1341
=0XC2550 + 0X1341
=0XC3891
Since both huge pointers p and q are pointing same physical address so if condition will true.

17.
What will be output if you will execute following c code?

```
#include<stdio.h>
int main(){
```

```
    char arr[7]="Network";
    printf("%s",arr);
   return 0;
}
```

7 char

(A) Network
(B) N
(C) network
✓(D) Garbage value
(E) Compilation error

Answer: (D)

18.
What will be output if you will execute following c code?

```
#include<stdio.h>
int main(){
    char arr[20]="MysticRiver";
    printf("%d",sizeof(arr));
    return 0;
}
```

✓(A) 20
(B) 11
(C) 12
(D) 22
(E) 24

Answer: (A)

enum

/*Error*/
enum x{
int a ;
char b ;
};
//we cannot add 'type', 'cause they're all int const. AND, we have to separate the members by

enum

/*Okay*/
enum x{
a = 5,
b = 'c'
};

/*Okay*/
enum x{
a , b
};

struct

/*Error*/
struct x{
int a = 5;
char b = 'c';
};
//we cannot assign value to the members in case of struct

19.
What will be output if you will execute following c code?

#1. An enumeration is a set of named integer constants. (you cant keep char or float or anything)

#2. Here fields/members are not declared with type.

#3. enum fields are separated by commas(,)

```
#include<stdio.h>
enum power{
    Dalai,
```

initially → 0

see, no type!

```
    Vladimir=3,
    Barack,
    Hillary
};
int main(){
    float leader[Dalai+Hillary]={1.f,2.f,3.f,4.f,5.f};
    enum power p=Barack;
    printf("%0.f",leader[p>>1+1]);
    return 0;
}
```

(A) 1
(B) 2
(C) 3
(D) 5
(E) Compilation error

Answer: (B)

20.
What will be output when you will execute following c code?

```
#include<stdio.h>
enum power{
    Dalai,
    Vladimir=3,
    Barack,
    Hillary
};
int main(){
    float leader[Dalai+Hillary]={1.f,2.f,3.f,4.f,5.f};
    enum power p=Barack;
    printf("%0.f",leader[p>>1+1]);
    return 0;
}
```

Choose all that apply:
(A) 1
(B) 2

(C) 3
(D) Compilation error
(E) None of the above

Answer: (B)

Explanation:
Size of an array can be enum constantan.
Value of enum constant Barack will equal to Vladimir + 1 = 3 +1 = 4
So, value of enum variable p = 4
leader[p >> 1 +1]
= leader[4 >> 1+1]
=leader[4 >> 2] //+ operator enjoy higher precedence than >> operator.
=leader[1] //4>>2 = (4 / (2^2) = 4/4 = 1
=2