1.
```
#include<stdio.h>
struct emp
{
    int age;
    struct emp *ptr;
};
int main(void)
{
    struct emp var={20,NULL};
    struct emp *ptr = &var;
    ptr->ptr = ptr;
    printf("%d %d",ptr->ptr->age,(*ptr).ptr->age);
}
```

A.20 20
B.20 NULL
C.0 0
D.Garbage Garbage

Answer: A

2.
```
#include <stdio.h>
#pragma pack(1)
int main(void)
{
    struct
    {
        int s[5];
        union
        {
            char a;  float b;
        }u1;
    } t;
    printf("%d", sizeof(t) + sizeof(t.u1));
    return 0;
}
```

A. 24
B. 28
C. 25
D. 20

Answer: B

3.
```
#include <stdio.h>
int main(void)
{
    struct test1
    {
        char name[15]; char *ptr;
    };
    struct test2
    {
        char *c ; struct test1 t1 ;
    };
    struct test2 t2 = {"Pune","Hinjawadi","Karad"};
    printf("%s%s\n",t2.c,t2.t1.ptr);
    printf("%s%s\n",++t2.c,++t2.t1.ptr);

    return 0;
}
```

A.PuneKarad
  unearad

B.PuneKarad
  PuneKarad

C.PuneKarad
  Garbage

D.Compile time Error

Answer: A

4.
```
#include <stdio.h>
struct test
{
    int data;
    struct test next;
};
int main(void)
{
    struct test t1;
    struct test t2;

    t1.data=100;
    t1.next=&t2;

    printf("%d%u",t1.data,t1.next);
    return 0;
}
```

A.100 2646865808(address of t2)
B.100 100
C.Compile time error
D.Abnormal termination

Answer:C

5.
```
#include <stdio.h>
struct test
{
    int data;
};
int main(void)
{
    struct test s1;
    s1.data=100;
    return 0;
}
```

If structure is declared as
follows & if i want to print value of data on console.
which of the following printf statements are valid

```
1. printf("%d",*(&s1.data));
2. printf("%d",s1.data);
3. printf("%d",*(&s1));
4. printf("%d",s1);
```

A. Only 2
B. Both 1 and 2
C. 1 2 and 3
D. All of above

Answers: D

6.

```
#include<stdio.h>
union testUnion
{
    short int num;
    char ch[2];
};
int main()
{
    union testUnion ut;
    ut.ch[0]=4;
    ut.ch[1]=8;
    printf("%d",ut.num);
    return 0;
}
```

A.2000
B.2052
C.1000
D.Garbage

Answer: B

7.
```
#include<stdio.h>
union test
{
    int i;
    char ch[2];
}u;
int main(void)
{
    union test var;
    var.i=256;

    printf("%d %d %d",var.i,var.ch[0],var.ch[1]);
    return 0;
}
```

A. 256 0 1
B. 256 0 0
C. Garbage values
D. 256 1 1

Answer: A

8.
```
#include<stdio.h>
int main(void)
{
    enum colors{RED=0,BLUE=-1,GREEN,YELLOW=-1};

    printf("%d, %d, %d, %d",RED,BLUE,GREEN,YELLOW);
    return 0;
}
```
A. Compile time Error
B. 0 -1 -2 -3 -1
C. 0, -1, 0, -1
D. 0,-1,-1,-1

Answer: C

```
9.
#include<stdio.h>
int main(void)
{
    typedef static int INT;
    INT a=100;
    printf("%d",a);
    return 0;
}
A. 100
B. 0
C. Compile time Error
D. No Output

Answer: C

10.
#include<stdio.h>
int main(void)
{
   enum colors{RED=0,BLUE,GREEN,YELLOW};
   enum location{MARKETYARD=-1,HINJAWADI,KARAD,SATARA};

   enum colors clr=RED;
   enum location lct=HINJAWADI;
    if(clr==lct)
        printf("True");
    else
        printf("False");
    return 0;
}

A. False
B. True
C. Compile time error
D. No output

Answer: B
```

11.
Enumeration constant is of

A. char type
B. Float type
C. int type
D. None of above

Answer: C

12.
```
#include<stdio.h>
int main(void)
{
    typedef int* INT;
    INT a;
    int b;
    printf("%d %d",sizeof(a),sizeof(b));
    return 0;
}
    // Note consider 64 bit compilation
```

A. 8 4
B. 4 4
C. 8 8
D. 4 8

Answer: A

13.
Amount of memory allocated to union object is

A. contain its largest member
B. contain its smallest member
C. The sum of memory required for all its member
D. None of these

Answer: A

```
14.
#include<stdio.h>
int main(void)
{
    typedef int INT;
    typedef INT SUNBEAM;
    SUNBEAM a=10;
    INT b=20;
    printf("%d %d",a,b);
    return 0;
}

A. Compile time error
B. Run time error
C. 10 20
D. 0 0

Answer: C

15.
#include<stdio.h>
#pragma pack(1)
struct Demo
{
    int bit1:1; int bit2:4; int bit3:3;
};
int main(void)
{
    struct Demo d1;
    printf("%d",sizeof(d1));
    return 0;
}
A. 1
B. 2
C. 8
D. 10

Answer: A
```

16.
```
#include<stdio.h>
struct Demo
{
    int num1:4;
    unsigned int num2:3;
    int num3:2;
};
int main(void)
{
    struct Demo d1;
    d1.num1=5;
    d1.num2=6;
    d1.num3=-1;
    printf("%d %d %d",d1.num1,d1.num2,d1.num3);

    printf("\n");
    d1.num1=10;
    d1.num2=9;
    d1.num3=3;
    printf("%d %d %d",d1.num1,d1.num2,d1.num3);

    return 0;
}
```

```
A. 5 6 -1
  -6 1 -1

B. 5 6  0
  -6 1  0

C. 5 6 -1
  -6 0  0

D. 5 6 -1
  -6-6-1
```

Answer: A