1.

```
#include<stdio.h>
int main()
{
    int a = 10;
    int *ptr = &a;
    printf("%d %d", *ptr,++*ptr);
    return 0;
}
```

A. 12 11
B. 10 11
C. 11 11
D. None of the above
Answer :C

2.
Find out the correct statement/s
1. void pointer can point to any type of location.
2. we need to typecast the pointer variable to dereference void pointer.

A. only 1 is correct
B. Only 2 is correct
C. both 1 and 2 are correct
D. both are incorrect
Answer :C

3.
Find out the correct statement
1. const int* ptr;
2. int * const ptr;

A. first statement declares pointer to a constant integer
B. first statement declares constant pointer to an integer
C. second statement declares pointer to an integer
D. second statement declares pointer to a constant integer

Answer :A

4.

```
#include<stdio.h>
int main()
{
    int a[] = {10,11};
    int* const ptr = a;
    *ptr = 11;
    printf("\n value at ptr is  : [%d]\n",*ptr);
    return 0;
}
```

A. 10
B. 11
C. Error
D. None
Answer :B

5.

```
#include<stdio.h>
int main()
{
    int a[5] = {1,2,3,4,5};
    int *ptr = (int*)(&a+1);
    printf("%d %d", *(a+1), *(ptr-1));
    return 0;
}
```

A. 2 5
B. 1 2
C. 4 5
D. 1 3
Answer :A

6.

```
#include<stdio.h>
int main()
{
    char *c[] = {"SunsQuiz", "MCQ", "TEST", "QUIZ"};
```

```
    char **cp[] = {c+3, c+2, c+1,
c};
    char ***cpp = cp;
    printf("%s ", **++cpp);
    printf("%s ", *--*++cpp+3);
    printf("%s ", *cpp[-2]+3);
    printf("%s ", cpp[-1][-1]+1);

    return 0;
}
```

A. TEST Quiz Z CQ
B. TEST sQuiz Z C
C. TEST sQuiz CQ
D. TEST sQuiz Z CQ
Answer :D

7.

```
#include<stdio.h>
int main()
{
    int k = 5;
    int *p = &k;
    int **m  = &p;
    **m = 6;
    ++*p++;
    printf("%d\n", k);
    return 0;
}
```

A. 6
B. 5
C. 7
D. None
Answer :C

8.

```
#include <stdio.h>
```

```
int main(void)
{
    int num1 = 10, num2 = 20, num3 = 30;
    int *ptr_num1 = &num1,    *ptr_num2 = &num2,
*ptr_num3 = &num3;
    int **spt = &ptr_num1;
    *spt = ptr_num2;
    printf("%d",*ptr_num1);

    return 0;
}
```

A. 20
B. 10
C. 30
D. Garbage value
Answer: A

9.
What is meaning of following declaration?

```
    int(*ptr[5])();
```

A. ptr is pointer to function.
B. ptr is array of pointer to function whoes return type is integer and who do not take any argument.
C. ptr is pointer to such function which return type is array.
D. ptr is pointer to array of function.

Answer :B

10.

```
#include <stdio.h>
int mul(int a, int b, int c)
{
    return a * b * c;
}
int main()
{
```

```
    int (function_pointer)(int,
int, int);
    function_pointer = mul;



    printf("The product of three numbers i%d " ,
            function_pointer(2, 3, 4));
    return 0;
}
```

A. Runtime error
B. compile time error
C. The product of three numbers is:24
D. None of the above
Answer :B

11.

```
#include <stdio.h>
void callvalue(int *x)
{
    x=x+10;
}
int main()
{
    int a=10;
    printf(" %d ",a);
    callvalue(&a);
    printf(" %d",a);
    return 0;
}
```

A. 10 10
B. 10 20
C. 20 20
D. None of the above
Answer :A

12.

```
#include<stdio.h>
int main( void )
{
    void *ptr_name=NULL;
    char ch=83, *name="SUNBEAM";
    int j=85;
    ptr_name=&ch;
    printf("%c", *(char*)ptr_name);
    ptr_name=&j;
    printf("%c", *(int*)ptr_name);
    ptr_name=name;
    printf("%s", (char*)ptr_name+2);
    return 0;
}
```

A. 83UNBEAM
B. S85NBEAM
C. SUNBEAM
D. NBEAM
Answer: C

13.

```
#include<stdio.h>
char* adding(char *value)
{
    value+=3;
    return(value);
}
int main(void)
{
    char *val1=NULL, *val2=NULL;
    val1 = "SUNBEAM";
    val2 = adding(val1);
    printf("Value = %s", val2);
    return 0;
}
```

A. Value = SUNBEAM
B. Value = EAM
C. Value = BEAM
D. Value = SUNB
Answer: C

14.

```
#include<stdio.h>
void changeVal(int *x)
{
    int i;
    for(i=0;i<sizeof(x);i++,x++)
    {
        *x+=2;
    }
}
int main()
{
    int arr[]={1,2,3,4,5,6,7};
    changeVal(arr);
    printf("%d %d %d",arr[4],arr[5],arr[6]);
    return 0;
}
```

A. Compiletime Error
B. 7 8 9
C. 5 6 7
D. None ofthe above
Answer: B

15.

```
#include<stdio.h>
int a()
{
    printf("Function");
    return 0;
}
```

```
int b()
{
    printf("Function in C");
    return 0;
}
int c()
{
    printf("C function");
    return 0;
}
int main()
{
    int (*ptr[3])();
    ptr[0] = a;
    ptr[1] = b;
    ptr[2] = c;
    ptr[2]();
    return 0;
}
```

A. Function
B. Function in C
C. C function
D. Error
Answer :C

16.

```
#include<stdio.h>
void abc(char arr[])
{
    printf("%c", *++arr);
    printf("%c", *arr++);
}
int main()
{
    char arr[100];
    arr[0] = 'a';
    arr[1] = 'b';
```

```
    arr[2] = 'c';
    arr[4] = 'd';
    abc(arr);
    return 0;
}
```

A. bb
B. bc
C. cc
D. cd
Answer :A