1.
```
#include<stdio.h>
signed char* convert(unsigned int *data)
{
    return ((signed char*)(data));
}
int main( void )
{
    unsigned int a = 577;
    signed char *data=NULL;
    data =convert(&a);
    printf(" %d ",*data);
    return 0;
}
```

A. 66
B. B
C. A
D. 65

Answer: D

2.

```
#include <stdio.h>
void calculate(float *ptr_floater1,float *ptr_floater2,
float *ptr_answer);
int main( void )
{
    float floater1 = 0.001, floater2= 0.003, answer=0.0f;
    calculate(&floater1,&floater2, &answer);
    printf("ans=%.3f", answer);
    return 0;
}
void calculate(float *ptr_floater1,float *ptr_floater2,
float *ptr_answer)
{
    *ptr_floater1 = 4 * *ptr_floater1;
    *ptr_answer= 3 * (*ptr_floater2 - *ptr_floater1);
    return;
}
```

```
A.  0.001
B. -0.003
C.  0.004
D.  0.003

Answer: B

3.
#include <stdio.h>
int x=10;
void callbyaddress(int *ptr_x)
{
    x=*ptr_x * *ptr_x / x;
}
int main( void )
{
    int x=100;
    printf(" x = %d ",x);
    callbyaddress(&x);
    printf(" x = %d",x);
    return 0;
}

A. x = 100   x = 100
B. x = 100    x = 10
C. x = 10     x = 10
D. Compile time error

Answer: A

3.

#include<stdio.h>
void callbyaddress(float *ptr_val)
{
    *ptr_val= 2.0f * *ptr_val;
    return;
}
```

```
int main( void )
{
    float val=3.14f;
    float * const strVal = &val;

    callbyaddress(&val);
    printf("%.1f",*strVal);
    return 0;
}
```

A. 3.1
B. 6.3
C. 6.2
D. compile time error

Answer: B

5.

```
#include <stdio.h>
void modify(int * const value)
{
    *value = 200;
    return ;
}
int main( void )
{
    const int value = 300;
    modify(&value);
    printf("value = %d\n", value);
    return 0;
}
```

A. value = 200
B. value = 300
C. garbage value
D. none of above

Answer: A

6.

```
#include <stdio.h>
int no=1000;
int* fun1(int *value)
{
    no += *value / *value;
    return &no;
}
int main( void )
{
    int num1=10;
    int *val=fun1(&num1);
    printf(" value= %d", *val);
    return 0;
}
```

A. value = 1001
B. value = 101
C. value = 1010
D. run time error

Answer: A

7.

```
#include<stdio.h>
int main( void )
{
    int num1 = 100;
    char ch='D';
    void *ptr_num1 = &num1;
    printf("%c-",--*(int*)ptr_num1);

    ptr_num1=&ch;
    printf("%c",--*(char*)ptr_num1);

    return 0;
}
```

```
A. 100 - D
B. 99 - C
C. c - C
D. D – D

Answer: C

8.
#include<stdio.h>
int main( void )
{
    void *ptr=NULL;
    char ch=72;
    int no='i';
    float f=4.46;
    ptr=&ch; printf("%c", *(char*)ptr);
    ptr=&no; printf("%c", *(int*)ptr);
    ptr=&f;  printf("-%.f", *(float*)ptr);
    return 0;
}

A. Hi-5
B. Hi-4
C. Hi-4.460000
D. Garbage value

Answer: B

9.
#include<stdio.h>
int main( void )
{
    int num1 = 100, num2=150;
    int *p = &num1, **pp = &p;
    **pp = num2;
    ++**pp;
    printf("num1=%d *p=%d **pp=%d\n", num1, *p, **pp);
    return 0;
}
```

```
A. num1=101 *p=101 **pp=101
B. num1=151 *p=151 **pp=151
C. num1=101 *p=100 **pp= garbage value
D. num1=151 *p=151 **pp= garbage value
```

Answer :B

10.

```
#include<stdio.h>
int main( void )
{
    int num1 = 100, num2=150;
    int *p = &num1;
    int **pp = &p;
    **pp = 75;
    --*p;
    ++**pp;
    num2=**pp;
    printf("num1=%d num2=%d ", num1, num2 );
    printf("*p=%d **pp=%d\n", *p, **pp );
    return 0;
}
```

```
A. num1=100 num2=150 *p=100 **pp=100
B. num1=100 num2=75 *p=74 **pp=75
C. num1=150 num2=150 *p=150 **pp=150
D. num1=75 num2=75 *p=75 **pp=75
```

Answer: D

11.

```
#include<stdio.h>
int fun(int **pp);
int main( void )
{
    int val = 10 ,*ptr1= &val;
    val=fun(&ptr1);
    printf(" val=%d *ptr=%d", val, *ptr1);
    return 0;
}
```

```
int fun(int **pp)
{
    **pp *= **pp ;
    printf(" **pp=%d ",**pp);
    return **pp/10;
}
```

```
A. **pp=100 val=10 *ptr=10
B. **pp=100 val=10 *ptr=100
C. **pp=100 val=100 *ptr=10
D. **pp=10 val=10 *ptr=10
```

Answer: A

12.

```
#include<stdio.h>
void changeVal2(int **x)
{
    **x/=2;
    return;
}
void changeVal1(int *x)
{
    *x*=5;
    return;
}
int main( void )
{
    int num1=10;
    int *ptr1=&num1;
    changeVal2(&ptr1);
    printf("num1=%d *ptr1=%d ",num1, *ptr1);

    changeVal1(ptr1);
    printf("num1=%d *ptr1=%d ",num1, *ptr1);

    return 0;
}
```

A. num1=10 *ptr1=5 num1=10 *ptr1=25
B. Run time error
C. num1=5 *ptr1=5 num1=25 *ptr1=25
D. Compile time error

Answer: C

13.

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

A. Compile time error
B. Run time error
C. a = 101 ptr = 101 a = 100 ptr = 100
D. a = 100 ptr = 101 a = 100 ptr = 99

Answer: C

14.

```
#include<stdio.h>
int main( void )
{
    const int a = 10 , b = 20 ;
    const int *  ptr = &a;
    int * const ptr1 = &a;
    printf("a = %d *ptr = %d ", a,*ptr);
    ptr = &b;
    *ptr1= b;
    printf("a = %d *ptr = %d ", a,*ptr);
    return 0;
}
```

```
A. a = 10 *ptr = 10 a = 10 *ptr = 20
B. a = 10 *ptr = 10 a = 20 *ptr = 20
C. compile time error
D. run time error

Answer: B

15.

#include<stdio.h>
int fun1(int n1, int n2)
{
    return n1+n2;
}

int fun2(int n2, int n1)
{
    return n1-n2;
}

void print(int n1, int n2,int (*fp)(int n1, int n2) )
{
    printf(" funptr = %d (*funptr) = %d" , fp, *fp);
    return;
}

int main( void )
{
    int val1=100, val2=200;
    int (*funptr)(int no1, int no2);

    funptr=fun1(val1,val2);
    print(val1, val2, funptr);

    funptr=fun2(val2, val1);
    print(10, 20, funptr);

    return 0;
}
```

A. run time error
B. funptr=300 (*funptr)=300 funptr= 100 (*funptr)= 100
C. funptr=300 (*funptr)=300 funptr=-100 (*funptr)=-100
D. compile time error

Answer: C

16.
What is the meaning of the statement?

```
int* fun(char (*a)[]);
```

A. fun is a function that accepts an argument which is a pointer to a character array returns a pointer to an integer.

B. fun is a function that accepts an argument which is an array of pointers to characters returns a pointer to an integer.

C. fun is a function that accepts an argument which is a pointer to a character array returns an integer.

D. fun is a function that accepts an argument which is an array of pointers to characters returns an integer.

Answer: A