1. value： if we need something actual value return to use or record.

   Reference: if we need to modify the value of variables which we had set

   Pointer:it is easy to create a area in heap to storage the data we need to release in hands,and it return pointer for some special situation which only pointer can use.

2. pass:it can protect the passed data pretend from being change;

   Return:when the return type is pointers or reference,it can protect the data from being modify.

3. int main(){

```
    Void print（int i）；
    Void print（char i）;
    Void print (double i);
}
```

```
4.. #include<iostream>
 #include <memory>
using namespace std;
int main()
{
int i=10;
auto_ptrap1(new int(4)),ap2;
ap2=ap1;
 cout << *ap2;
 cout << *ap1 << endl;
char *c; shared_ptr sc(new char(10));
c=sc.get();
return 0;
}
```

5.

```
625
18.4
6.6
9409
9
18.0
```

6.

```
hello world
h
hello
```

The second be error

Use RAII (Resource Acquisition Is Initialization) principle: Prefer using objects and classes that automatically manage dynamic memory through their constructors and destructors, such as std::vector, std::string, and smart pointers like std::unique_ptr and std::shared_ptr.Avoid raw pointers: Use smart pointers, such as std::unique_ptr and std::shared_ptr, to manage dynamic memory. They provide automatic memory deallocation and help prevent memory leaks and dangling pointers.

```
7. Int min(int a[],int length)
{
    Int min=a[0];
    For(int i=1;i<length;i++)
    {
        If(a[i]<ans)
        {
            Min=a[i];
        }
    }

}
8. double min(double a[],int length)
{
    double min=a[0];
    For(int i=1;i<length;i++)
    {
        If(a[i]<ans)
        {
            Min=a[i];
        }
    }

}
9. float min(int a[],int length,double flag)
{
    float min=a[0];
    For(int i=1;i<length;i++)
    {
        If(a[i]<ans)
        {
            Min=a[i];
        }
    }

}
10. long min(int a[],int length,long flag)
{
    long min=a[0];
    For(int i=1;i<length;i++)
    {
        If(a[i]<ans)
        {
            Min=a[i];
        }
    }

}
```