```
//Generic(or void pointer)
#include <iostream>
using namespace std;
int main()
{
    int a=10;
    float b=34.56f;
    char c='A';
    void *ptr=&a;
    cout<<*(int*)ptr<<endl;
    ptr=&b;
    cout<<*(float*)ptr<<endl;
    ptr=&c;
    cout<<*(char*)ptr;
    return 0;
}
//Constant pointer
#include <iostream>
using namespace std;
int main()
{
    int a=10,b=20;
    int *const ptr=&a;
    cout<<*ptr;
    ptr=&b;//Compile time error as ptr is constant pointer
    return 0;
}
//Constant pointer
#include <iostream>
using namespace std;
int main()
{
    int a=10,b=20;
    int *const ptr=&a;
    cout<<*ptr<<endl;
    //ptr=&b;//Compile time error as ptr is constant pointer[We cannot change the address hold by pointer]
    *ptr=100;//We can change the value of the variable towards which pointer is pointing
    cout<<*ptr;
    return 0;
}
//Pointer to constant
#include <iostream>
using namespace std;
int main()
{
    int a=10,b=20;
    const int *ptr=&a;
    cout<<*ptr<<endl;
    ptr=&b;
    cout<<*ptr;
    *ptr=100;//Compile time error will come as we cannot modify the value
    return 0;
}
//Constant pointer to constant
#include <iostream>
using namespace std;
int main()
{
    int a=10,b=20;
    const int *const ptr=&a;
    cout<<*ptr<<endl;
    //ptr=&b;//Compile time error as ptr is constant pointer[We cannot change the address hold by pointer]
    //*ptr=100;//Compile time error ,We cannot change the value of the variable towards which pointer is
pointing
    cout<<*ptr;
    return 0;
}
//Dangling pointer(Compile time memory allocation case
#include <iostream>
using namespace std;
int main()
{
    int *ptr;
    {
        int a=10;
        ptr=&a;
        cout<<*ptr<<endl;
        cout<<ptr<<endl;
    }
    cout<<ptr;//Before and after variable goes out of scope ptr is having same memory address[Hence it is
dangling pointer]
    return 0;
}
//Dangling pointer(Compile time memory allocation case) with solution
#include <iostream>
using namespace std;
int main()
{
    int *ptr;
    {
        int a=10;
        ptr=&a;
        cout<<*ptr<<endl;
        cout<<ptr<<endl;
    }
    ptr=NULL;
    cout<<ptr;//Zero will be printed(No longer a dangling pointer)
    return 0;
}
```