INDEX :

Pointer New And Delete Keyword :

```cpp
#include<iostream>

using namespace std;


int main(){

    int a = 4;

    int *ptr = &a;

    cout<<"The value of 'a' is : "<<*(ptr)<<endl;

    // 'new' Operator

    float *p = new float(40);

    cout<<"The address at p is : "<<p<<endl;

    cout<<"The value at the address p is : "<<*p<<endl;


    int *arr = new int[3];

    arr[0] = 10;

    *(arr + 1) = 20;  // We can write it like this also.

    arr[2] = 30;

    cout<<"The value of arr[0] is : "<<arr[0]<<endl;

    cout<<"The value of arr[1] is : "<<arr[1]<<endl;

    cout<<"The value of arr[2] is : "<<arr[2]<<endl;


    // 'delete' Operator

    delete p;  // Use to delete previous data in p

    cout<<"The value at the address p is : "<<*p<<endl; // This will give grabage value


    delete[] arr;  // Use to delete previous data in arr[]

    arr[0] = 11;

    arr[1] = 12;

    arr[2] = 13;

    cout<<"The value of arr[0] is : "<<arr[0]<<endl;

    cout<<"The value of arr[1] is : "<<arr[1]<<endl;

    cout<<"The value of arr[2] is : "<<arr[2]<<endl;


return 0;

}
```

Output :-

```
PS D:\9. Tutorial of C++> cd "d:\9. Tutorial
The value of 'a' is : 4
The address at p is : 0xf66d88
The value at the address p is : 40
The value of arr[0] is : 10
The value of arr[1] is : 20
The value of arr[2] is : 30
The value at the address p is : 40
The value of arr[0] is : 11
The value of arr[1] is : 12
The value of arr[2] is : 13
PS D:\9. Tutorial of C++>
```

Arrow Operator :

```cpp
#include<iostream>

using namespace std;


class Complex{

    int real, imaginary;

    public:

    void setData(int a, int n){

        real = a;

        imaginary = n;

    }

    void getData(){

        cout<<"The value of real part is : "<<real<<endl;

        cout<<"The value of imaginary part is : "<<imaginary<<endl;

    }

};


int main(){

    Complex c1;

    Complex *ptr = &c1;

    // Complex *ptr = new Complex; // We can write like this also

    // (*ptr).setData(3, 4); // is exactily same as below:

    ptr->setData(3, 4); // '->' this is Arrow operator.

    (*ptr).getData();


return 0;

}
```

Output :-

```
PS D:\9. Tutorial of C++> cd "d:\9. Tutorial
The value of real part is : 3
The value of imaginary part is : 4
PS D:\9. Tutorial of C++>
```

Array Of Object Using Pointer :

```
#include<iostream>
using namespace std;
class Shop{
    int Id;
    float price;
    public:
    void setData(int a, int n){
        Id = a;
        price = n;
    }
    void getData(){
        cout<<"Code of this item is : "<<Id<<endl;
        cout<<"And price of this item is : "<<price<<endl;
    }
};
int main(){
    int size, x, y;
    cout<<"Enter the size of the object here : "<<endl;
    cin>>size;
    Shop *ptr = new Shop[size];
    Shop *ptrTemp = ptr;
    for(int i=0; i<size; i++){
        cout<<"Enter the id and price of the item number "<<i+1<<" respectively"<<endl;
        cin>>x>>y;
        ptr->setData(x, y);
        ptr++;
    }
    for(int i=0; i<size; i++){
        cout<<"Item Number : "<<i+1<<endl;
        ptrTemp->getData();
        ptrTemp++;
    }
return 0;
}
```

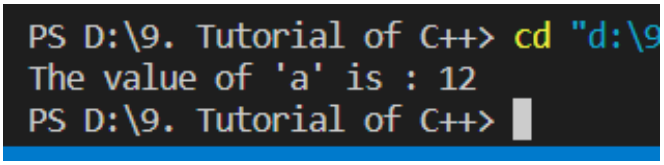
Output :-

```
PS D:\9. Tutorial of C++> cd "d:\9. Tutorial of C++\" ; if
inter }
Enter the size of the object here :
2
Enter the id and price of the item number 1 respectively
101
1500
Enter the id and price of the item number 2 respectively
102
1300
Item Number : 1
Code of this item is : 101
And price of this item is : 1500
Item Number : 2
Code of this item is : 102
And price of this item is : 1300
PS D:\9. Tutorial of C++> 2
```

this Pointer :

```cpp
#include<iostream>

using namespace std;


class A{

    int a;

    public:

    /* void setData(int a){

        a = a; // This will give garbage value, and wouldn't give any error also.

    } */


    void setData(int a){

    /* 'this' is a keyword which is a pointer which points to the objects which

    invokes the member function. */

        this->a = a;

    }

    /* A & setData(int a){

        this->a = a;

        return *this; // We can use like this also, in return statement.

    } */

    void getData(){

        cout<<"The value of 'a' is : "<<a<<endl;

    }

};


int main(){

    A a;

    a.setData(12);

    a.getData();


    // a.setData(21).getData(); --> We can write like this also as setData returns a value.


return 0;

}
```

Output :-



Polymorphism :

```cpp
#include<iostream>

using namespace std;


/* Polymorphism in C++ can be of two types :

1. Compile time polymorphism

   1.1 - Function Overloding

   1.2 - Operator Overloding


2. Run time polymorphism

   2.1 - Virtual function

*/


class Base{

    public:

    int varBase;

    void display(){

        cout<<"Displaying base class variable varBase : "<<varBase<<endl;

    }

};


class Derived : public Base{

    public:

    int varDerived;

    void display(){

        cout<<"Displaying base class variable varBase : "<<varBase<<endl;

        cout<<"Displaying derived class variable varBase : "<<varDerived<<endl;

    }

};


int main(){

    Base *BCPointer;

    Base objBase;


    Derived objDerived;
```

```
    BCPointer = &objDerived; // Pointing base class pointer to derived class


    BCPointer->varBase = 34;
    // BCPointer->varDerived = 134; --> This will give error.
    BCPointer->display();  // Its a example of late binding


    Derived *DCPointer;
    DCPointer = &objDerived;
    DCPointer->varBase = 21; // We can access it as it is inherited from base class.
    DCPointer->varDerived = 12;
    DCPointer->display();


return 0;
}
```

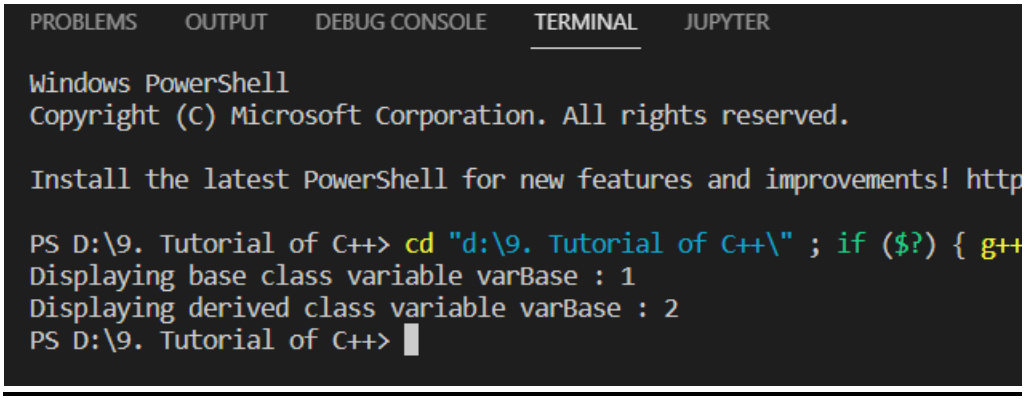
Output :-

```
PS D:\9. Tutorial of C++> cd "d:\9. Tutorial of C++\" ;
Displaying base class variable varBase : 34
Displaying base class variable varBase : 21
Displaying derived class variable varBase : 12
PS D:\9. Tutorial of C++>
```

Virtual Function :

```
#include<iostream>

using namespace std;

/* Rules for virtual functions :-

1. They cannot be static.

2. They are accessed by object pointers.

3. Virtual functions can be a friend of another class.

4. A virtual function in the base class might not be used.

5. If a virtual function is defined in the base class, there is no necessity of

   redefining it in the derived class.

*/

class Base{

    public:

    int varBase = 1;

    virtual void display(){ // Because of virtual fuction we can access Derived function.

        cout<<"Displaying base class variable varBase : "<<varBase<<endl;

    }

};

class Derived : public Base{

    public:

    int varDerived = 2;

    void display(){

        cout<<"Displaying base class variable varBase : "<<varBase<<endl;

        cout<<"Displaying derived class variable varBase : "<<varDerived<<endl;

    }

};

int main(){

    Base *BCPointer;

    Derived objDerived;


    BCPointer = &objDerived;

    BCPointer->display();


return 0;

}
```

Output :-


PROBLEMS
OUTPUT
DEBUG CONSOLE
TERMINAL
JUPYTER
Windows PowerShell
Copyright (C) Microsoft Corporation. All rights reserved.
Install the latest PowerShell for new features and improvements! http
PS D:\9. Tutorial of C++> cd "d:\9. Tutorial of C++\" ; if ($?) { g++
Displaying base class variable varBase : 1
Displaying derived class variable varBase : 2
PS D:\9. Tutorial of C++>

Abstract Base Class And Pure Virtual Function :

```cpp
#include<iostream>

using namespace std;


class CWMe{

    protected:

    string s;

    float rate; // This whole class is act as a 'Abstract' base class.

    public:

    CWMe(string str, float rts) : s(str), rate(rts){}

    virtual void display() = 0; // Do nothing function --> Pure virtual function

};


class Video : public CWMe{

    float videoTime;

    public:

    Video(string st, float rt, float vt) : CWMe(st, rt){

        videoTime = vt;

    }

    void display(){

        cout<<"The tittle of the video is : "<<s<<endl;

        cout<<"The rating is : "<<rate<<endl;

        cout<<"Time length of the video is : "<<videoTime<<endl;

    }

};


class Text : public CWMe{

    int words;

    public:

    Text(string st, float rt, int wr) : CWMe(st, rt){

        words = wr;

    }

    void display(){

        cout<<"The tittle of the text is : "<<s<<endl;

        cout<<"The rating is : "<<rate<<endl;
```

```
        cout<<"Number of word in the text is : "<<words<<endl;
    }
};

int main(){
    string s;
    float l, r;
    int w;
    s = "Its me ";
    r = 4.19;
    l = 5.89;
    CWMe *obj1[2];

    Video objVideo(s, r, l);
    obj1[0] = &objVideo;
    obj1[0]->display();

    s = "Its you ";
    r = 3.59;
    w = 120;
    Text objText(s, r, w);
    obj1[1] = &objText;
    obj1[1]->display();

return 0;
}
```

Output :-

```
PS D:\9. Tutorial of C++> cd "d:\9. Tutorial
ndPureVF }
The tittle of the video is : Its me
The rating is : 4.19
Time length of the video is : 5.89
The tittle of the text is : Its you
The rating is : 3.59
Number of word in the text is : 120
PS D:\9. Tutorial of C++>
```