### Pointers Vs Reference Variables

| **BASIS FOR COMPARISON** | **POINTER** | **REFERENCE** |
| --- | --- | --- |
| Basic | The pointer is the memory address of a variable. | The reference is an alias for a variable. |
| Returns | The pointer variable returns the value located at the address stored in pointer variable which is preceded by the pointer sign '\*'. | The reference variable returns the address of the variable preceded by the reference sign '&'. |
| Operators | \*, -> | & |
| Null Reference | The pointer variable can refer to NULL. | The reference variable can never refer to NULL. |
| Initialization | An uninitialized pointer can be created. | An uninitialized reference can never be created. |
| Time of Initialization | The pointer variable can be initialized at any point of time in the program. | The reference variable can only be initialized at the time of its creation. |
| Reinitialization | The pointer variable can be reinitialized as many times as required. | The reference variable can never be reinitialized again in the program. |





**Generic Pointers**

When a variable is declared as being a pointer to type **void** it is known as a *generic pointer*. Since we cannot have a variable of type **void**, the pointer will not point to any data and therefore cannot be dereferenced. It is still a pointer though, to use it we just have to cast it to another kind of pointer first. Hence the term *Generic pointer*. This is very useful when we want a pointer to point to data of different types at different times.

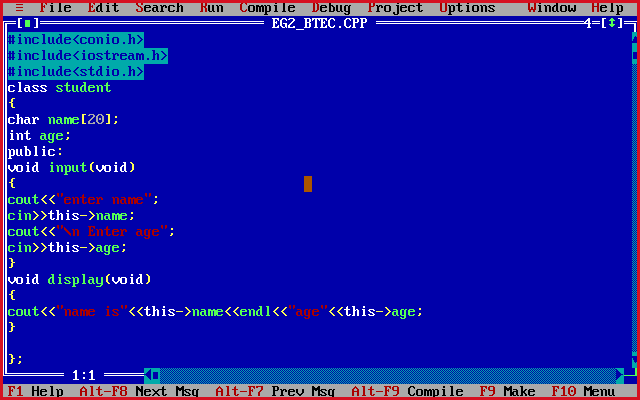


**This Pointer**

To understand ‘this’ pointer, it is important to know how objects look at functions and data members of a class.

1. Each object gets its own copy of the data member.
2. All-access the same function definition as present in the code segment.

Meaning each object gets its own copy of data members and all objects share a single copy of member functions.  
Then now question is that if only one copy of each member function exists and is used by multiple objects, how are the proper data members are accessed and updated?  
The compiler supplies an implicit pointer along with the names of the functions as ‘this’.  
The ‘this’ pointer is passed as a hidden argument to all nonstatic member function calls and is available as a local variable within the body of all nonstatic functions. ‘this’ pointer is not available in static member functions as static member functions can be called without any object (with class name).



*Pointer to object*

