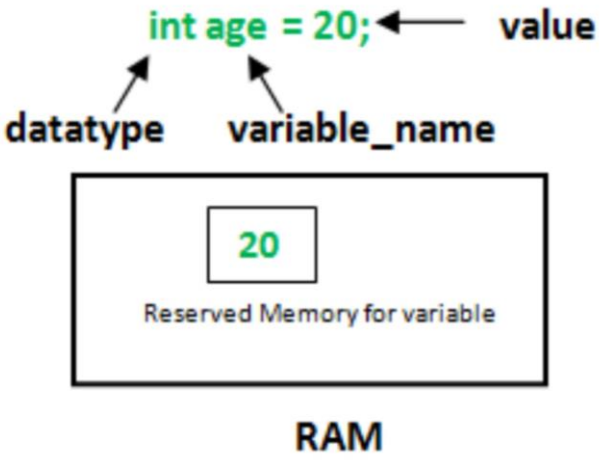
Techimax

Fundamentals of Programming in C++

Introduction to Pointers

What is a variable ?

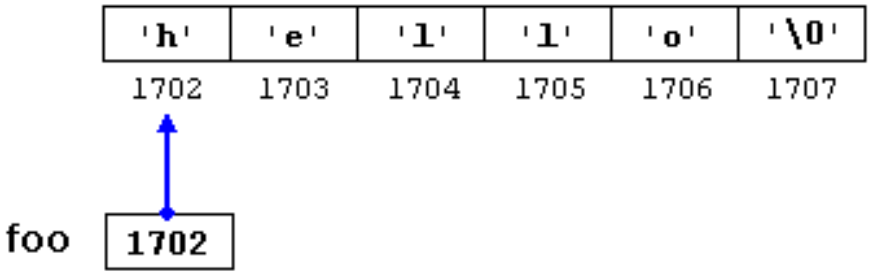
A variable is a name for a piece of memory that holds a value




```
int a [ ] = {10,20,30,40};

int *p[4];    // *p[0], *p[1], *p[2], *p[3]
```

2886728	2886732	2886736	2886740
10	20	30	40
a[0]	a[1]	a[2]	a[3]



Introduction to Pointers

The address-of operator (&)
See what memory address is assigned to the variable

The dereference operator (*)
The dereference operator (*) allows us to access the value at a particular address.

```
int x = 12;
cout << x << '\n';
// 12
cout << &x << '\n';
// 0×61ff0c
cout << *(&x) << '\n';
// 12
```

Pointer
A pointer is a variable that holds a *memory address* as its value.

Pointers

Declaring a Pointer

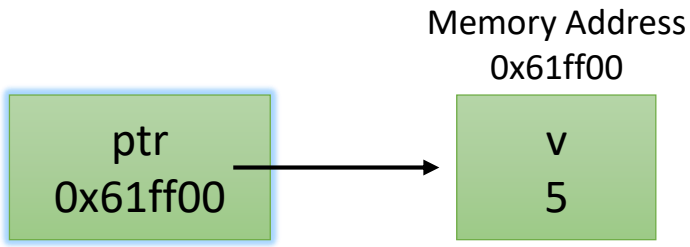
```
// valid & recommended
int *iPtr;
float *fPtr;
int* ptfFunction()

// valid but not recommended
int * iPtr2;
int* iPtr3;

// iPtr4 is a pointer
// iPtr5 is an int
int *iPtr4, iPtr5;
```

Assigning a value to a Pointer

```
int v = 5;
int *ptr = &v;
int *ptr2;
ptr2 = &v;
```



Pointers

Dereferencing Pointers

A dereferenced pointer evaluates to the *contents* of the address it is pointing to.

```
int var = 5;
cout << &var << " " << var << endl;
// 0x61ff08 5
int *ptr = &var;
cout << ptr << " " << *ptr << endl;
// 0x61ff08 5
```

Pointers

Arrays & Pointers

```
int ar[5] = {1, 5, 8, 9, 0};
cout << ar << " " << *ar << endl;
for (int i = 0; i < 5; i++)
{
    cout << ar[i] << " " << *(ar + i) << " "
      << &ar[i] << " " << ar + i << endl;
}
```

```
0x61fef8 1
1 1 0x61fef8 0x61fef8
5 5 0x61fefc 0x61fefc
8 8 0x61ff00 0x61ff00
9 9 0x61ff04 0x61ff04
0 0 0x61ff08 0x61ff08
```

```
char name[] = "Jenny";
cout << *name << " " <<
  *(name + 1) << *(name + 4) << endl;
```

```
J ey
```

Pass by Value vs Pass by Reference

Pass by Value

```
void func(int x)
{
    x = 10;
    cout << "value of x from func : " << x << endl;
}

int main()
{
    //  pass by value
    int x = 5;
    func(x);
    cout << "value of x from main : " << x << endl;
}
```

```
value of x from func : 10
value of x from main : 5
```

Pass by Value vs Pass by Reference

Pass by Reference

```
void func(int &x)
{
    x = 10;
    cout << "value of x from func : " << x << endl;
}

int main()
{
    // pass by reference
    int x = 5;
    func(x);
    cout << "value of x from main : " << x << endl;
}
```

```
value of x from func : 10
value of x from main : 10
```

Conditional or Ternary Operator (?:)

Short Hand if-else

```
variable = Expression1 ? Expression2 : Expression3
```

```
int getMax(int a, int b)
{
    return (a > b) ? a : b;
}

int main()
{
    int a = 5, b = 10;
    cout << "Max num is : " << getMax(a, b);
}
```