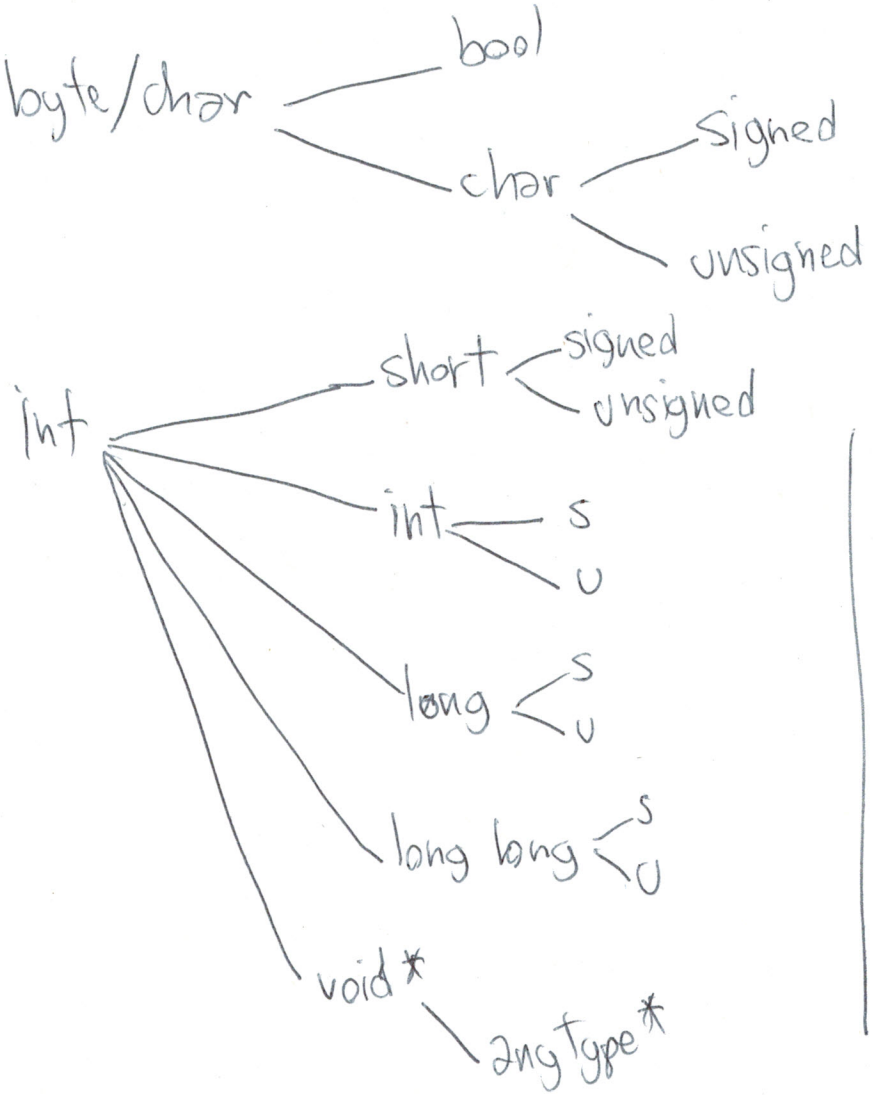
Or

```
cout << *value;
Cout.operator<< (value);
```

1.5

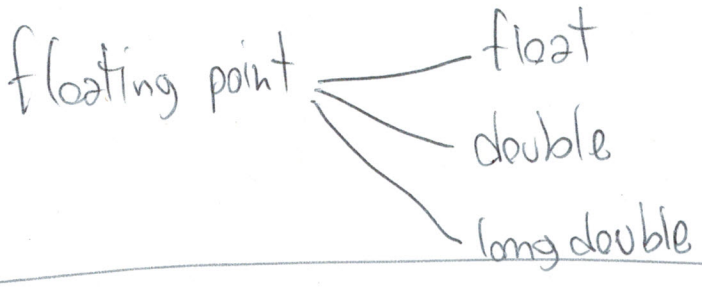
FUNCTION
- DECLARE
- DEFINITION

```
int toCelsius (int farhenheitValue);

int toCelsius (int f) {
    return (f-32)x5/9;
}

    int toCelsius (double f){

    }
```



long abc;

long int abc;



1

11

1.0

1.13.1

```
int main (int argc, char * argv[]){
...
return value;
}
```

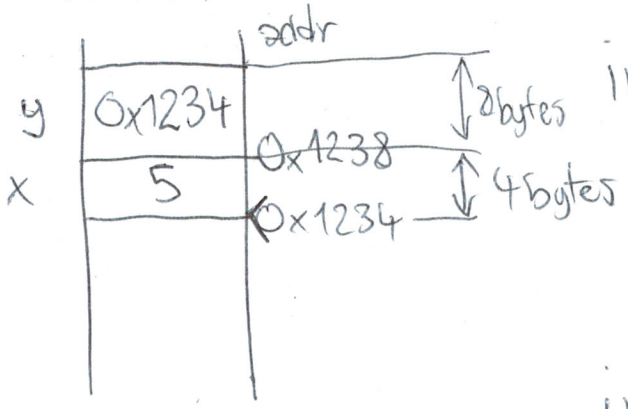
1.14

```
const int x=33;
x = x+1; ← COMPILER ERROR
```

1.15

UNARY OPERATORS → & address-of
UNARY OPERATORS → * dereference

MEMORY




```
int x=5;
int * y = &x;
```

```
int counter;
void inc(int c){
    c++;
}
...
int main(){
    inc(counter);
}
```

```
void inc(int * c){
    *c++;
}
... int main(){
    inc(&c);
}
```

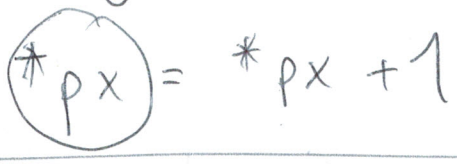
int *

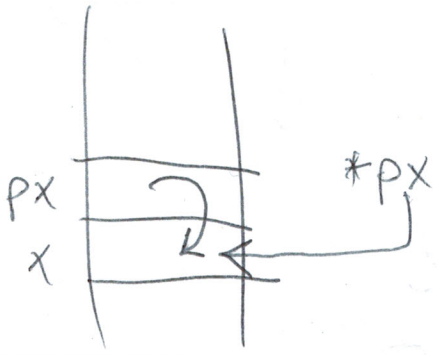
int **

int ***

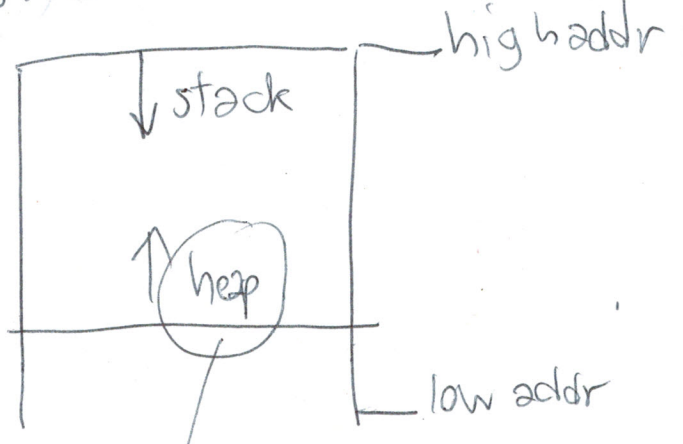
---

x = x + 1





1.15.2



new —— malloc

delete —— free

```
int foo () {
    int j=13;
```

1.16 REFERENCE VARIABLE

C++ ONLY

LVALUE

```
int n;
int& rn = n;
```

2.2

```
class ClassName {
public:
    public Members;
private:
    private Members;
protected:
    protected Members;
}
```

public: — ANYWHERE

private: — ONLY CLASS MEMBERS