const

foo() const	no change private variable
foo(const &)	no change the val passed in

class

do not forget ;	class Name(){\\ };
default constructor	Name();
main function arguments	int main(int argc, char* argv[])
error message	if (!inFile) {std::cerr << “ERROR!!” << std::endl;}
class name .h file	Name::getValue(){}
read file	

```
ifstream puzzleStr(argv[1]);
string puzzleLine;
vector<string> puzzleText;
while (puzzleStr.good())
{
puzzleStr >> puzzleLine;
puzzleText.push_back(lower(puzzleLine));
}
```

Pointers

new	delete([]) a;
int * p, q;	p is int pointer, q is int
a[0]	pointer of array a[]
a[i]; all equal to	*(a+i)
array is a pointer	p[3] = *(p+3)
int list[10]; int*p=list;	p = list[0];
p = &x;	take the address of x to p
*p = 13;	follow the pointer
*p = a; all equal to	p = &a;
Delete an array of pointer	

```
for (int i = 0; i < rows; i++)
delete [] a[i];
delete [] a;
```

dynamic memory allocation

Automatic memory	allocates space on the stack and stores there
Static memory	They are not eliminated when they go out of scope
Dynamic memory	explicitly allocated (on the heap) as needed
new	new TYPE;
loop delete	only for two heaps;