

Subscript Operators & Pointers

•Subscript operator: array

- Date darray [5];
- Date d1 = darray [3];

•Subscript operator: pointer

- Accesses element in array
- Uses "pointer arithmetic"

Pointers: Subscript Equivalent

```
Date da1[10];
Date da2[10];
```

← Create arrays and access elements

```
da1[3] = da2[5];
```

← Create pointers

```
Date* pa1 = &da1[0];
Date* pa2 = da2;
```

← Access elements by subscripting pointers

```
pa1[3] = pa2[5];
```

← Equivalent form

```
*(pa1+3) = *(pa2+5);
```

← Equivalent form



Pointer Arithmetic

Left operand	Operator	Right operand	Result
Pointer	+	Number	Pointer
Number	+	Pointer	Pointer
Pointer	-	Number	Pointer
Pointer	-	Pointer	Number

Pointer(s) must be in same memory block!

Pointer Arithmetic Example

```

int ar[6];
int* pa = ar;
int* pb = pa + 3;

*pa = 47;   pa: [ ] → 0: 47
*pb = 32;   pb: [ ] → 3: 32
pa[1] = 22;
pb[1] = 64;
*(pa+5) = 9;

```

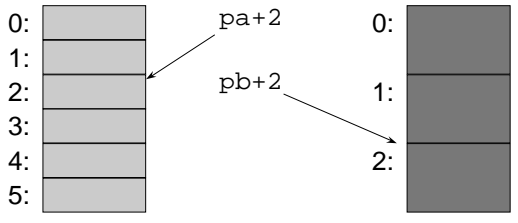
0:	47
1:	22
2:	
3:	32
4:	64
5:	9

Object Size Effects

```

int* pa = ar; // int ar[6];
Date* pb = dt; // Date dt[3];

```

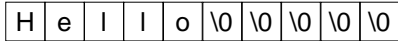


C-Style Strings

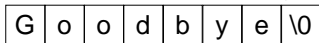
- **Character (char) arrays**
 - Characters are array elements
 - End marked by zero (NUL) character
- **C-string library support**
 - In <cstring>
 - Routines require careful use
 - See text, page 622

C-String Initialization

```
char s1 [10] = "Hello";
char s2 [] = "Goodbye";
```

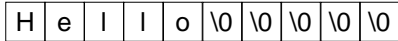


NUL terminator

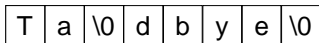


C-String Input/Output

```
cout << s1; // "Hello";
cin >> s2; // "Goodbye";
```



Output: "Hello"

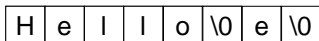
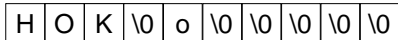


User types "Ta ta for now"

What if user types "ABCDEFGH"?

C-String Assignment

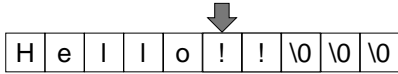
```
strcpy (s2, s1); // array
strcpy (s1+1, "OK"); // literal
```



Destination must be long enough!

C-String Concatenation

```
strcat (s1, "!!"); // append
```



Scan to end of destination string
Append source characters at end

Destination must be long enough!

C-String Comparison



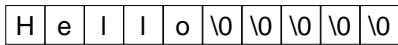
```
int val = strcmp (s1, s2);
```

Compare strings, character at a time
Return value: 0 if equal up to '\0'
<0 if s1 less than s2
>0 if s1 greater than s2

Strings must be NUL-terminated!

C-String Length

```
char s1 [10] = "Hello";  
int len = strlen(s1);
```



len = 5



C-String Manipulation

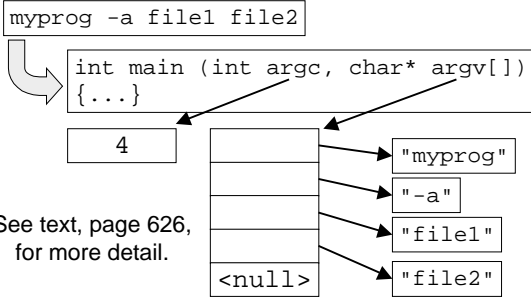
```

char s1 [10] = "Hello";
MakeUpper (s1); // char* ??

void MakeUpper (char* str)
{
  char* cp = str;
  while (*cp) // until '\0'
  {
    *cp = toupper (*cp);
    cp++;
  }
}

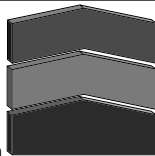
```

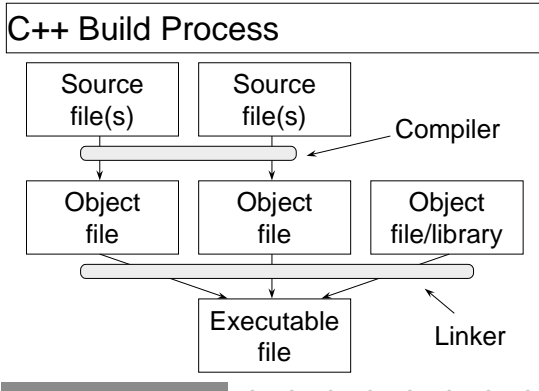
Command Line Parameters

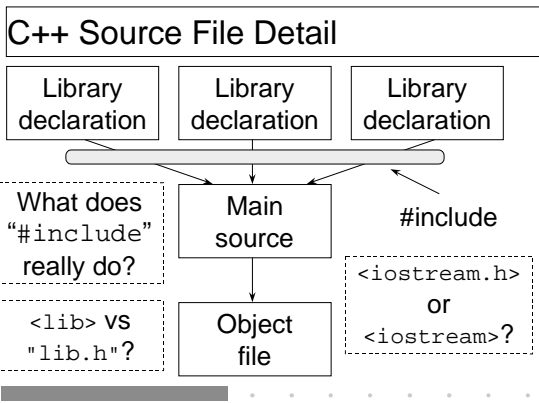


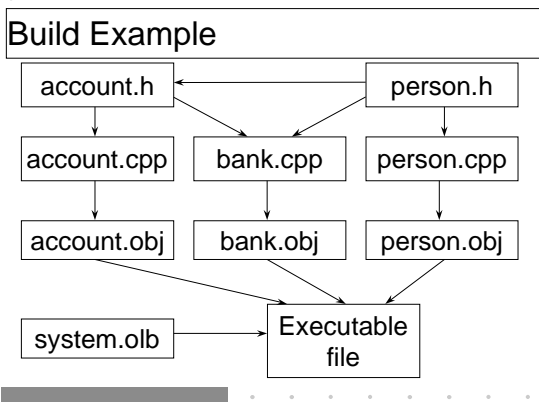
C++ Program Components

- **Main program**
 - main(), other functions
- **Class libraries**
 - Interface and implementation
- **System libraries**
 - Interface and implementation

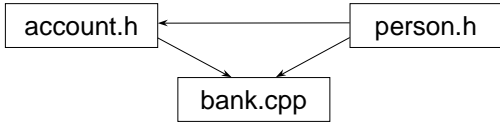








Duplicate Definition Problem



bank.cpp includes person.h
 bank.cpp includes account.h
 account.h includes person.h

So person.h gets included twice!
 But duplicate class definition is an error . . .

Idempotence

- **Multiple includes possible**
 - Of one library interface (header file)
- **But want only one definition**
 - Of classes defined in interface
- **How do we do it?**
 - Directives (now) or automatic (future?)

Idempotence with Directives (#)

<p>person.h:</p> <p><u>First include:</u> PERSON_H undefined (a "macro") Define PERSON_H Define class Person</p> <p><u>Subsequent includes:</u> PERSON_H is defined Definitions not executed</p>	<pre> #ifndef PERSON_H #define PERSON_H class Person { ... }; #endif </pre>
--	---
