| UCLA |
|------|
|------|

| CS 31:   |  |
|--|--|
| Introduction To Computer Science I HOWARD A. Stahl                                 |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| Student s( "Sam Smith", 104604956 );<br>Student * ptrS;                            |  |
| <pre>ptrS = &amp;s<br/>ptrS = new Student( "Sally Jones", 204705987 );</pre>       |  |
| Student array[ 5 ];  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| Student s( "Sam Smith", 104604956 );   |  |
| Student * ptrS;<br>ptrS = &s   |  |
| <pre>ptrS = new Student( "Sally Jones", 204705987 );<br/>Student array[ 5 ];</pre> |  |
|  |  |
|  |  |
|  |  |

| A Few Odds And E | inds |
|------------------|------|

## Student s( "Sam Smith", 104604956 ); Student \* ptrs; ptrs = &s; ptrs = new Student( "Sally Jones", 204705987 ); Student array[ 5 ];



#### A Few Odds And Ends...

### Student s( "Sam Smith", 104604956 ); Student \* ptrS; ptrS = &s;

ptrS = &s; ptrS = new Student( "Sally Jones", 204705987 ); Student array[ 5 ];





#### A Few Odds And Ends...

#### Student s( "Sam Smith", 104604956 );

Student \* ptrS; ptrS = &s; ptrS = new Student( "Sally Jones", 204705987 ); Student array[ 5 ];





| A Few Odds And Ends  |  |
|--|--|
|  |  |
| Student s( "Sam Smith", 104604956 ); Student * ptrS;                               |  |
| <pre>ptrS = &amp;s ptrS = new Student( "Sally Jones", 204705987 );</pre>           |  |
| Student array[ 5 ];  |  |
|  |  |
| Allocates Stack Or   |  |
| Memory? YES Heap? Stack  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| A Few Odds And Ends  |  |
| Student s( "Sam Smith", 104604956 );   |  |
| <pre>Student * ptrS; ptrS = &amp;s</pre>   |  |
| <pre>ptrS = new Student( "Sally Jones", 204705987 );<br/>Student array[ 5 ];</pre> |  |
| Student array[ 5 ],  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| 45 OH 4 IS I   |  |
| A Few Odds And Ends  |  |
| Student s( "Sam Smith", 104604956 );   |  |
| <pre>Student * ptrS; ptrS = &amp;s</pre>   |  |
| <pre>ptrS = new Student( "Sally Jones", 204705987 );</pre>                         |  |
| Student array[ 5 ];  |  |
|  |  |
| Allocates<br>Memory?   |  |
|  |  |

| A Few Odds And Ends  |  |
|--|--|
|  |  |
| Student s( "Sam Smith", 104604956 ); Student * ptrS;                               |  |
| <pre>ptrS = &amp;s ptrS = new Student( "Sally Jones", 204705987 );</pre>           |  |
| Student array[ 5 ];  |  |
|  |  |
| Allocates<br>Memory?   |  |
| NO   |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Farry Odda And Finds   |  |
| A Few Odds And Ends  |  |
| Student s ( "Sam Smith", 104604956 );  |  |
| Student * ptrS;<br>ptrS = &s   |  |
| <pre>ptrS = new Student( "Sally Jones", 204705987 );<br/>Student array[ 5 ];</pre> |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| Student s( "Sam Smith", 104604956 );   |  |
| Student * ptrS;<br>ptrS = &s   |  |
| <pre>ptrS = new Student( "Sally Jones", 204705987 );<br/>Student array[ 5 ];</pre> |  |
|  |  |
| Allocates  |  |
| Memory?  |  |
|  |  |

| A Few Odds And Ends  |  |
|--|--|
|  |  |
| Student s( "Sam Smith", 104604956 );<br>Student * ptrS;                            |  |
| ptrS = &s  |  |
| <pre>ptrS = new Student( "Sally Jones", 204705987 );<br/>Student array[ 5 ];</pre> |  |
|  |  |
| Allocates  |  |
| Memory?  |  |
| NO   |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| 71, 61, 61, 61, 61, 61, 61, 61, 61, 61, 6  |  |
| Student s( "Sam Smith", 104604956 );<br>Student * ptrS;                            |  |
| ptrS = &s  |  |
| <pre>ptrS = new Student( "Sally Jones", 204705987 ); Student array[ 5 ];</pre>     |  |
| beddene dridje o 1,  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| 2. 1   |  |
| Student s( "Sam Smith", 104604956 ); Student * ptrS;                               |  |
| <pre>ptrS = &amp;s ptrS = new Student( "Sally Jones", 204705987 );</pre>           |  |
| Student array[ 5 ];  |  |
|  |  |
| Allocates  |  |
| Memory?  |  |
|  |  |
|  |  |

| A Few Odds And Ends  |  |
|--|--|
| <pre>Student s ( "Sam Smith", 104604956 ); Student * ptrs; ptrs = &amp;s</pre>               |  |
| <pre>ptrS = new Student( "Sally Jones", 204705987 );<br/>Student array[ 5 ];</pre>           |  |
|  |  |
| Allocates<br>Memory?<br>YES  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| Student s( "Sam Smith", 104604956 ); Student * ptrS;   |  |
| <pre>ptrS = &amp;s ptrS = new Student( "Sally Jones", 204705987 ); Student array[ 5 ];</pre> |  |
| Student array( ) 1,  |  |
| Allocates<br>Memory? Stack Or<br>Heap?   |  |
| YES  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| Student s( "Sam Smith", 104604956 );   |  |
| <pre>Student * ptrS; ptrS = &amp;s ptrS = new Student( "Sally Jones", 204705987 );</pre>     |  |
| Student array[ 5 ];  |  |

| A Few Odds And Ends  |  |
|--|--|
| , , , , , , , , , , , , , , , , , , ,                      |  |
| Student s( "Sam Smith", 104604956 );                       |  |
| Student * ptrS;  |  |
| ptrS = &s  |  |
| ptrS = new Student( "Sally Jones", 204705987 );            |  |
| Student array[ 5 ];  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
|  |  |
| Student s( "Sam Smith", 104604956 );                       |  |
| Student * ptrS;<br>ptrS = &s                               |  |
| ptrs = new Student( "Sally Jones Assuming );               |  |
| Student array[ 5 ];  Student() Resolves                    |  |
| Resolves   |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| A Lew Odds Alld Ellds                                      |  |
| Student s( "Sam Smith", 104604956 );                       |  |
| Student * ptrS;  |  |
| ptrS = &s  |  |
| <pre>ptrS = new Student( "Sally Jones", 204705987 );</pre> |  |
| Student array[ 5 ];  |  |
|  |  |
|  |  |
| Allocates<br>Memory?                                       |  |
|  |  |
| •  |  |

| Δ | Few  | 2PPU | $\Delta nd$ | Ends  |
|---|------|------|-------------|-------|
| А | LEAN | Ouus | Allu        | LIIUS |

```
Student s( "Sam Smith", 104604956 );
Student * ptrS;
ptrS = &s;
ptrS = new Student( "Sally Jones", 204705987 );
Student array[ 5 ];
```



#### A Few Odds And Ends...

```
Student s( "Sam Smith", 104604956 );
Student * ptrS;
ptrS = &s;
ptrS = new Student( "Sally Jones", 204705987 );
Student array[ 5 ];
```





#### A Few Odds And Ends...

Student s( "Sam Smith", 104604956 );
Student \* ptrS;
ptrS = &s;
ptrS = new Student( "Sally Jones", 204705987 );
Student array[ 5 ];





| A Few Odds And Ends                     |  |
|---|--|
| Student * ptrArray;                     |  |
| <pre>ptrArray = new Student[ 5 ];</pre> |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
| A Few Odds And Ends                     |  |
| Student * ptrArray;                     |  |
| <pre>ptrArray = new Student[ 5 ];</pre> |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
| A Few Odds And Ends                     |  |
| Student * ptrArray;                     |  |
| Allocates Memory?                       |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |

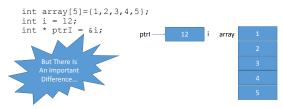
| A Few Odds And Ends   |  |
|---|--|
| <pre>Student * ptrArray; ptrArray = new Student[ 5 ];</pre> |  |
|   |  |
| Allocates   |  |
| Memory?<br>NO   |  |
|   |  |
|   |  |
|   |  |
|   |  |
| A Few Odds And Ends   |  |
| <pre>Student * ptrArray; ptrArray = new Student[ 5 ];</pre> |  |
| perarray - new Student[ 3 ],                                |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
| A Few Odds And Ends   |  |
| Student * ptrArray;   |  |
| <pre>ptrArray = new Student[ 5 ];</pre>                     |  |
|   |  |
|   |  |
|   |  |

| A Few Odds And Ends  |  |
|--|--|
| Student * ptrArray;  |  |
| <pre>ptrArray = new Student[ 5 ];</pre>                                      |  |
|  |  |
| Allocates<br>Memory?   |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| <pre>Student * ptrArray; ptrArray = new Student[ 5 ];</pre>                  |  |
| Allocates  |  |
|  |  |
| Memory?<br>YES   |  |
|  |  |
|  |  |
|  |  |
|  |  |
| A Few Odds And Ends  |  |
| Student * ptrArray;  |  |
| ptrArray = new Student[ 5 ];  Allocates Memory? YES  Allocates Memory? Heap? |  |
|  |  |
|  |  |
|  |  |

| A Few Odds And Ends  |  |
|--|--|
| ATEW Gads And Ends   |  |
| <pre>Student * ptrArray; ptrArray = new Student[ 5 ];</pre>          |  |
|  |  |
|  |  |
| Allocates Stack Or Heap?   |  |
| YES Heap   |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| Equivalencies  |  |
| Arrays Are Pointer Variables   |  |
| <pre>int array[5]={1,2,3,4,5}; int i = 12; int * ptrI = &amp;i</pre> |  |
| inc - pcii - αi,   |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| Equivalencies  |  |
| Arrays Are Pointer Variables   |  |
| int array[5]={1,2,3,4,5};  |  |
| int i = 12;<br>int * ptrI = &i ptrI — 12 i array 1                   |  |
| 2  |  |
| 4  |  |
| 5  |  |

#### Equivalencies

Arrays Are Pointer Variables



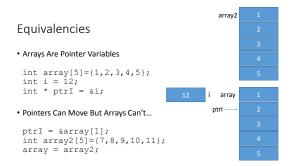
#### Equivalencies

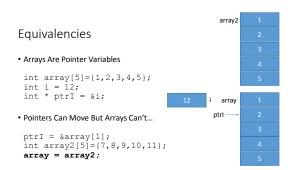
Arrays Are Pointer Variables

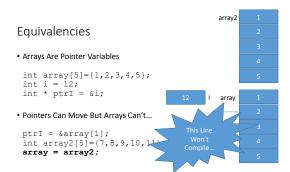
```
int array[5]={1,2,3,4,5};
int i = 12;
int * ptrI = &i;
```

• Pointers Can Move But Arrays Can't...

```
ptrI = &array[1];
int array2[5]={7,8,9,10,11};
array = array2;
```







#### Equivalencies

- Arrays Are Pointer Variables
- Pointers Can Move But Arrays Can't Move Because They Are Fixed Pointers
- The Arrow Cannot Move (Be Changed)
- $\bullet$  The Box It Points To Change Be Changed However...
- int array[ 5 ] Is Implemented As A: int \* const array

| The Issue Of const  |  |
|---|--|
| • const Always Means "Fixed"  |  |
| The Question Is "What Are We Fixing In"   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
| The Issue Of const  |  |
| <ul> <li>const Always Means "Fixed"</li> <li>The Question Is "What Are We Fixing In"</li> </ul> |  |
| Example #1:   |  |
| const int NUM_MONTHS=12;<br>const double SALARY=12.50;  |  |
| Control deduce Grande 12.00)  |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
| The Issue Of const  |  |
| • const Always Means "Fixed"  |  |
| • The Question Is "What Are We Fixing In"   |  |
| Example #1:   |  |
| const int NUM_MONTHS=12; const double SALARY=12.50; The Named Values Are Fixed                  |  |
| And Cannot Be<br>Changed.   |  |
|   |  |
| ·   |  |

| The Issue Of const   |  |
|--|--|
| • const Always Means "Fixed"   |  |
| The Question Is "What Are We Fixing In"  |  |
| Example #1:  |  |
| const int NUM_MONTHS=12; const double Salary=12,50; The Named Values Are Fixed  The Dathber in t |  |
| And Cannot Be Changed  Changed  Changed  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| The Issue Of const   |  |
| <ul><li>const Always Means "Fixed"</li><li>The Question Is "What Are We Fixing In"</li></ul>   |  |
|  |  |
| Example #1: Example #2:  const int NUM_MONTHS=12; void foo(const int & i); const double SALARY=12.50;  |  |
| Course double Salant-12.30,  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
| The Issue Of const   |  |
| • const Always Means "Fixed"   |  |
| The Question Is "What Are We Fixing In"  |  |
| Example #1: Example #2:  |  |
| <pre>const int NUM_MONTHEF12;</pre>  |  |
| Value Is Fixed And Cannot Be Changed   |  |
| Change   |  |

| The Issue Of co   | nst  |  |   |   |
|---|--|--|---|---|
| • const Always Means  |  |  |   | _ |
| • The Question Is "What                                     |  |  |   | _ |
| Example #1:  const int NUM MONTHS=12;                       | <pre>Example #2:     void foo( const int &amp; i);</pre> |  |   |   |
| const double SALARY=12,507 The Parameter i's Value Is Fixed |  | const Applies To                           |   |   |
| And Cannot Be Changed                                       |  | The Datatype                               |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
| The Issue Of co   | nst  |  | - |   |
| • const Always Means  |  |  | - | _ |
| The Question Is "What                                       | Are We Fixing In"  |  |   | _ |
| <pre>Example #1: const int NUM_MONTHS=12;</pre>             | <pre>Example #2: void foo( const int &amp; i);</pre>     | Example #3:                                |   |   |
| const double SALARY=12.50;                                  | void 100( const int & i);                                | <pre>public:<br/>int getFoo() const;</pre> |   |   |
|   |  | <pre>private:   int mFoo; };</pre>         |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
|   |  |  |   |   |
| The Issue Of co   | nst  |  |   | _ |
| • const Always Means  | ((F)   |  |   |   |

 $\bullet$  The Question Is "What Are We Fixing In..."

getFoo Is Read-Only And Cannot Change Any Member Variable Values... Example #3:

Example #1:

const int

#### The Issue Of const

- const Always Means "Fixed"
- The Question Is "What Are We Fixing In..."



#### The Issue Of const

• When Working With Pointers, What Can We Mark const?

int i = 12;  
int \* ptrI = &i ptrI 
$$\longrightarrow$$
 12 i

#### The Issue Of const

• When Working With Pointers, What Can We Mark const?

int i = 12;  
int \* ptrI = &i ptrl 
$$\longrightarrow$$
 12 i

• We Could Mark The Arrow const

#### The Issue Of const

• When Working With Pointers, What Can We Mark const?

int i = 12;  
int \* ptrI = &i 
$$ptrl \rightarrow 12$$
 i

- We Could Mark The Arrow const
  - In Which Case ptrI Cannot Be Pointed Anywhere Else...
  - In Which Case int j = 13; ptrI = &j; Would Be Illegal

#### The Issue Of const

• When Working With Pointers, What Can We Mark const?

int i = 12;  
int \* const ptrI = &i ptrl 
$$\longrightarrow$$
 12 i

- We Could Mark The Arrow const
  - In Which Case ptrI Cannot Be Pointed Anywhere Else...
  - In Which Case int j = 13; ptrI = &j; Would Be Illegal

#### The Issue Of const

• When Working With Pointers, What Can We Mark const?

int 
$$i = 12$$
;  
int \* const ptrI = &i ptrI  $\longrightarrow$  12 i

- We Could Mark The Arrow const
  - In Which Case ptrI Cannot Be Pointed Anywhere
  - In Which Case int j = 13; ptrI =



| The  | Issue  | $\bigcap f$ | ~    | net  |
|------|--------|-------------|------|------|
| 1116 | 122115 | ()I         | (:(: | 1151 |

• When Working With Pointers, What Can We Mark const?

| int | i | = 12;  |    |      |   |
|-----|---|--------|----|------|---|
|     |   | ptrI = | &i | ptrl | i |

• We Could Mark The Box const

#### The Issue Of const

• When Working With Pointers, What Can We Mark const?

int i = 12;  
int \* ptrI = &i ptrl 
$$\longrightarrow$$
 12 i

- We Could Mark The Box const
  - In Which Case i Cannot Be Changed Via ptrI...
  - In Which Case \*ptrI = 13; Would Be Illegal

#### The Issue Of const

• When Working With Pointers, What Can We Mark const?

- We Could Mark The Box const
  - In Which Case i Cannot Be Changed Via  ${\tt ptrI...}$
  - In Which Case \*ptrI = 13; Would Be Illegal

| The Issue Of const  |  |
|---|--|
| When Working With Pointers, What Can We Mark const?   |  |
| int i = 12;<br>const int * ptrI = &i ptrl — 12 i  |  |
|   |  |
| We Could Mark The Box const     In Which Case i Cannot Be Changed Via ptrI     Const Applies To |  |
| • In Which Case *ptrI = 13; Would Be n. The Datatype int  |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
| Question 5 Hint   |  |
| <pre>char msg[100] = "UC Los Angeles"; deleteCapitals(msg);</pre>                               |  |
| <pre>cout &lt;&lt; msg &lt;&lt; endl;</pre>   |  |
| • Should Print: " os ngeles"  |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
|   |  |
| Question 5 Hint   |  |
| <ul><li>void deleteCapitals( char * data );</li></ul>   |  |
|   |  |
| UC Los Angeles \0   |  |
| 1   |  |
| data  |  |
|   |  |
|   |  |

# Question 5 Hint •void deleteCapitals( char \* data ); data Question 5 Hint •void deleteCapitals( char \* data ); Question 5 Hint •void deleteCapitals( char \* data ); Copy Over To



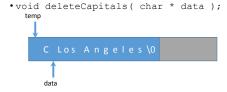
#### Question 5 Hint

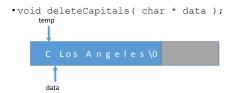


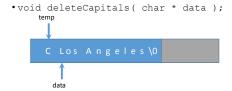




#### Question 5 Hint







#### Question 5 Hint





| $\sim$ |          | _ |   |     |
|--------|----------|---|---|-----|
| ( )ı   | Jestion. | _ | ы | ını |
|        |          |   |   |     |





- One Very Important Final Point!
- Be Sure To Copy Over The Ending NULL Character To Be Sure The Data You Produce Is A Valid C-String