



Systems Programming Concepts



- Structures
- . Typedef
- Declarations
- . Using Structures with Functions
- . Structure Example



10.1 Introduction

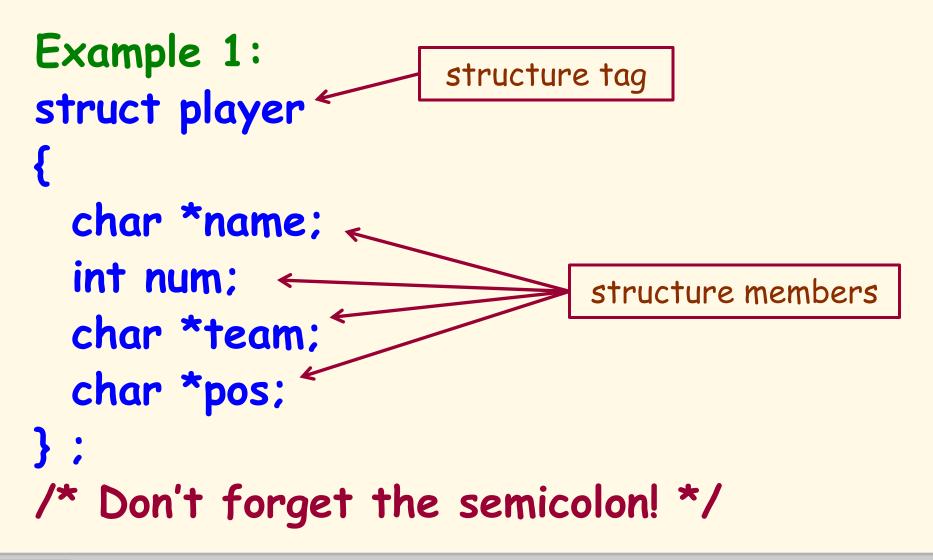
. Structures

- A collection of related variables aggregated under one name.
 - Can contain variables of different data types.
- Commonly used to define records to be stored in files.
- *When combined with pointers, structures can create linked lists, stacks, queues, and trees.

© 2007 Pearson Ed -All rights reserved.



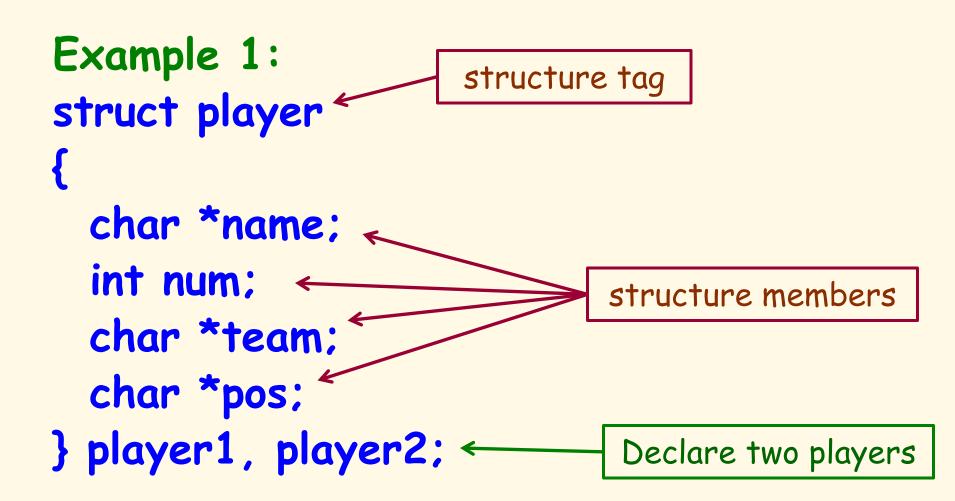
Structures





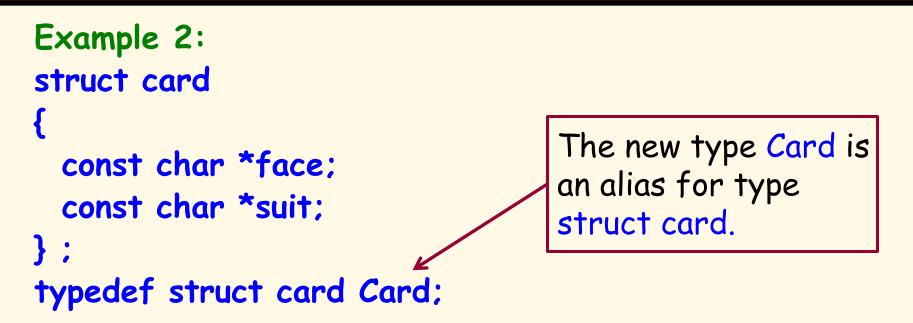


Structures





Typedef Example



struct introduces the definition for structure card.

- card is the structure name and is used to declare variables of the structure type.
- card contains two members of type char *
 - These members are face and suit.





Typedef Example

```
Another way to declare this!!
typedef struct
 const char *face:
 const char *suit:
} Card;
```

Card deck[52];

© 2007 Pearson Ed -All rights reserved.



10.6 typedef

Example:

typedef struct Card *CardPtr;

or

Card *Cardptr;

- Defines a new type name CardPtr as an alias for type struct Card *.
- typedef does not create a new data type.
 - It only creates an alias.
- Capitalize the first letter of typedef names to emphasize that they are synonyms for other type names.

© 2007 Pearson Ed -All rights reserved.



10.2 Structure Definitions

struct information

- A struct cannot contain an instance of itself.
- It can contain a member that is a pointer to the same structure type (a self-referential structure).
- A structure definition does not reserve space in memory.
 Rather a struct creates a new data type used to define structure variables.
- · Definitions
 - Defined like other variables:
 - card oneCard, deck[52], *cPtr;
 - Can use a comma separated list:

struct card {
 char *face;
 char *suit;

} oneCard, deck[52], *cPtr;

© 2007 Pearson Ed -All rights reserved.



10.2 Structure Definitions

- Valid Operations

- Assigning a structure to a structure of the same type.
- Taking the address (&) of a structure
- Accessing the members of a structure.
- Using the sizeof operator to determine the size of a structure.



^{© 2007} Pearson Ed -All rights reserved.

10.3 Initializing Structures

- Initializer lists
 - Example:

struct card oneCard = { "Three", "Hearts" };

- Assignment statements
 - Example:

struct card threeHearts = oneCard;

- Could also define and initialize threeHearts as follows:

struct card threeHearts; threeHearts.face = "Three"; threeHearts.suit = "Hearts";



10.4 Accessing Members of Structures

- Accessing structure members
 - The dot operator (.) {the structure member operator} is used to access a structure member via the structure variable name.

```
card myCard;
printf( "%s", myCard.suit );
```

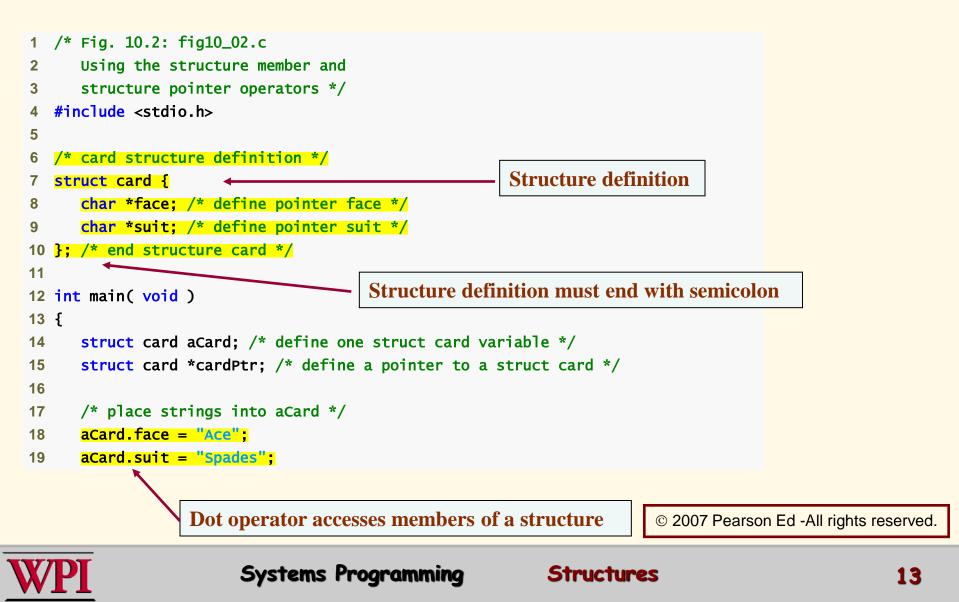
 The arrow operator (->) {the structure pointer operator} accesses a structure member via a pointer to the structure.

```
card *myCardPtr = &myCard;
printf( "%s", myCardPtr->suit );
```

- myCardPtr->suit is equivalent to
 - (*myCardPtr).suit



Structure member and pointer operators



Structure member and pointer operators

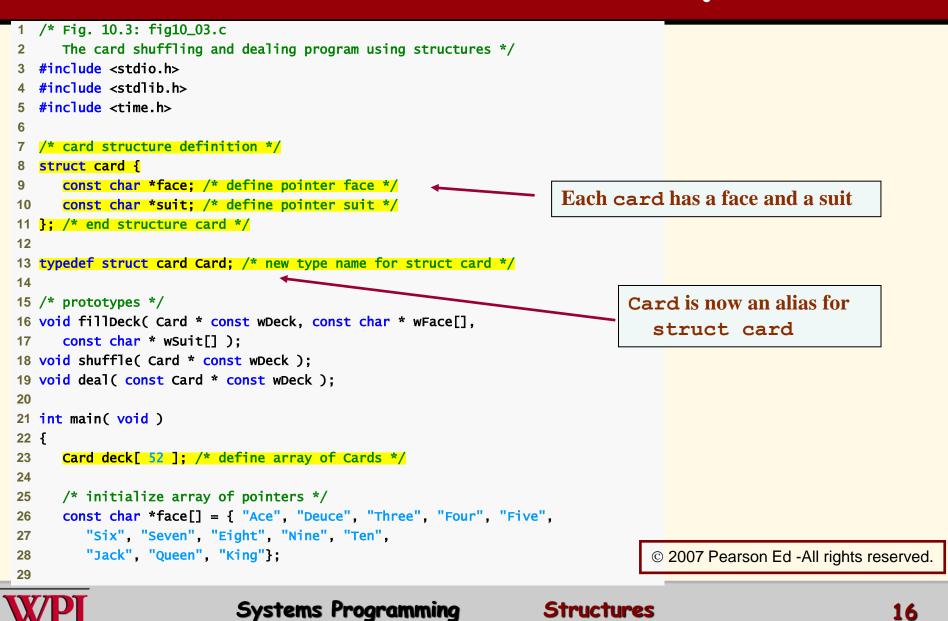
20	
21 cardPtr = &	aCard; /* assign address of aCard to cardPtr */
22	
23 printf("%s	%s%s\n%s%s%s\n%s%s%s\n", aCard.face, " of ", aCard.suit,
24 cardPtr-	<pre>>face, " of ", cardPtr->suit,</pre>
25 (*cardP	<pre>Ptr).face, " of ", (*fardPtr).suit);</pre>
26	
27 return 0; /	<pre>/* indicates successful termination */</pre>
28	
29 } /* end main	*/
Ace of Spades	
Ace of Spades	
Ace of Spades	
	Arrow operator accesses members
	of a structure pointer



10.5 Using Structures with Functions

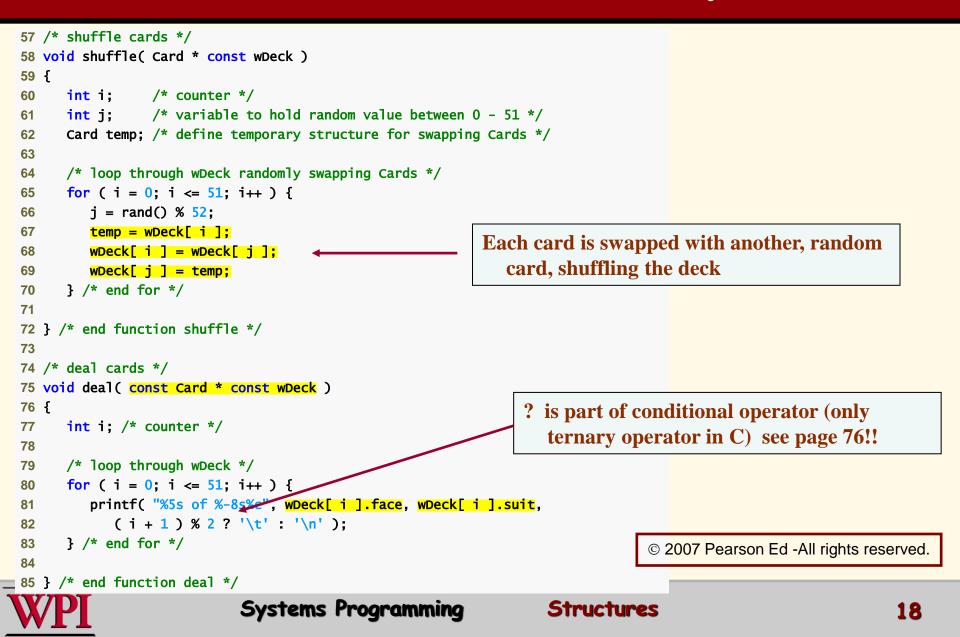
- Passing structures to functions
 - The entire structure can be passed.
 - Individual members of the structure can be passed.
 - For both cases, they are passed by value.
- . To pass a structure by-reference
 - Pass the address of the structure variable.
- . To pass arrays by-value
 - Create a structure with the array as a member and then pass the st<u>ructure.</u>





```
/* initialize array of pointers */
30
      const char *suit[] = { "Hearts", "Diamonds", "Clubs", "Spades"};
31
32
      srand( time( NULL ) ); /* randomize */
33
34
      fillDeck( deck, face, suit ); /* load the deck with Cards */
35
      shuffle( deck ); /* put Cards in random order */
36
      deal( deck ); /* deal all 52 Cards */
37
38
      return 0; /* indicates successful termination */
39
40
41 } /* end main */
42
43 /* place strings into Card structures */
44 void fillDeck( Card * const wDeck, const char * wFace[],
      const char * wSuit[] )
45
46 {
                                               Constant pointer to modifiable array
      int i; /* counter */
47
                                                  of Cards
48
      /* loop through wDeck */
49
      for ( i = 0; i <= 51; i++ ) {</pre>
50
         wDeck[ i ].face = wFace[ i % 13 ];
51
                                                       Fills the deck by giving each
         wDeck[ i ].suit = wSuit[ i / 13+];
52
                                                          Card a face and suit
      } /* end for */
53
54
55 } /* end function fillDeck */
                                                                    © 2007 Pearson Ed -All rights reserved.
56
```





Four of Clubs Three of Diamonds Four of Diamonds Nine of Hearts Three of Clubs Eight of Clubs Deuce of Clubs Seven of Clubs Ace of Clubs Ace of Spades Seven of Diamonds Eight of Spades Five of Spades Queen of Spades Oueen of Diamonds Jack of Diamonds Eight of Hearts King of Spades Eight of Diamonds Ace of Hearts Four of Spades Deuce of Hearts Deuce of Spades Seven of Spades King of Clubs Ten of Hearts

Three of Hearts Three of Spades Ace of Diamonds Ten of Clubs Four of Hearts Nine of Diamonds **Oueen of Clubs** Jack of Spades Five of Diamonds Five of Clubs Six of Spades **Oueen of Hearts** Deuce of Diamonds Six of Hearts Seven of Hearts Nine of Spades Five of Hearts Six of Clubs Ten of Spades King of Hearts Jack of Hearts Jack of Clubs Ten of Diamonds Nine of Clubs Six of Diamonds King of Diamonds



Review of Structure

- Definition of structures in C
- Syntax details for declaring structs
- . Initializing structs
- . Typedef
- Structure member (.) and pointer -> operators
- Passing structures to functions
- . A Structure Example

