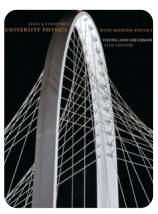
CH 14 QUIZ

Share

14 studiers today Leave the first rating

Textbook solutions for this set

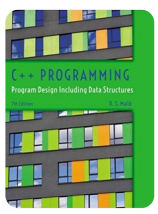


Sears and Zemansky's University Physics with Modern Physics

13th Edition • ISBN: 9780321696861 (6 more)

Hugh D. Young, Lewis Ford, Roger A. Freedman

7,837 solutions



C++ Programming: Program Design Including Data Structures

7th Edition • ISBN: 9781285852751 (1 more)

D. S. Malik

867 solutions

Search for a textbook or question

Terms in this set (38)

[1401] Which of these lines correctly prints 3?
 	
struct S {
int a = 3;
double b = 2.5;
};
 	
S obj, *p = &obj;
 	
cout << p.a << endl;
cout << *p.a << endl;
cout << *(p).a << endl;
cout << *(p.a) << endl;
cout << (*p).a << endl;
	cout << (*p).a << endl;

[1402] Which of these lines correctly prints 2.5?
 	
struct S {
int a = 3;
double b = 2.5;
};
 	
S obj, *p = &obj;
 	
cout << *(p).b << endl;
cout << *p.b << endl;
cout << p->b << endl;
cout << *(p.b) << endl;
cout << *p->b << endl;
	cout << p->b << endl;

[1403] Which of these lines displays the eighth element of a?
 	
int a[15];
 	
cout << a[8] << endl;
cout << a(7) << endl;
cout << a.at(7) << endl;
cout << a[7] << endl;
	cout << a[7] << endl;

int a[] = {1, 2, 3};
 	
cout << a.length << endl;
cout << sizeof(a[0]) << endl;
cout << a.size() << endl;
cout << sizeof(a) << endl;
None of these	

[1405] What is stored in the last element of nums? int nums[3] = {1, 2}; Undefined value 2 Syntax error in array declaration 0 1	0
[1406] Which line throws and out_of_range exception? double speed[5] = {. . .}; None of these cout << speed[4] << endl; cout << speed[5] << endl; cout << speed[0] << endl; cout << speed[1] << endl;	None of these
[1407] Which line has undefined output? double speed[5] = {. . .}; cout << speed[5] << endl; cout << speed[0] << endl; None of these cout << speed[1] << endl; cout << speed[4] << endl;	cout << speed[5] << endl;
[1408] Which line creates an array with 5 elements? int[5] d; int b[5]; int a[4]; None of these int[] c[5];	int b[5];

[1409] What is printed?
 	
int a[] = {1, 2, 3};
int b[] = {1, 2, 3};
 	
if (a == b) cout << "a == b" << endl;
else cout << "a != b" << endl;
 	
a != b
Undefined behavior
a == b
Syntax error; does not compile.
	a != b

int a[] = {1, 2, 3}; int b[] = {4, 5, 6}; a = b; Syntax error; does not compile. {4, 5, 6} {1, 2, 3} Undefined behavior	
[1411] Which assigns a value to the first position in letters? char letters[26]; letters[0] = 'a'; letters[0] = "a"; letters[1] = 'b'; letters.front() = 'a'; letters = 'a';	letters[0] = 'a';
[1412] Which assigns a value to the first position in letters? char letters[26]; *letters = 'a'; *letters = "a"; *letters[0] = 'a'; *(letters + 1) = 'a'; *letters + 1 = 'b';	*letters = 'a';

[1413] What does this loop do?
 	
int a[] = {6, 1, 9, 5, 1, 2, 3};
int x(0);
for (auto e : a) x += e;
cout << x << endl;
 	
Counts the elements in a
Selects the largest value in a
Has no effect
Selects the smallest value in a
Sums the elements in a
	Sums the elements in a

[1414] What is the address of the first pixel in the last row of this image?
 	
Pixel *p; // address of pixel data
int w, h; // width and height of image
 	
p + w + h
p + w + (h - 1)
p + w * h
p + w * (h - 1)
None of these are correct
	p + w * (h - 1)

Pixel *p; // address of pixel data
int w, h; // width and height of image
 	
*p + w - 1
None of these are correct
*(p + w) - 1
p + w - 1
*(p + w - 1)	

[1416] Which returns the last pixel on the first row of this image?
 	
Pixel *p; // address of pixel data
int w, h; // width and height of image
 	
p[w - 1]
*p[w - 1]
None of these are correct
p[w] - 1
p + w - 1
	p[w - 1]

[1417] What is the equivalent array notation? int dates[10]; cout << (*dates + 2) + 2 << endl; dates[0] + 4 dates[2] + 2 dates[2] dates[0] + 2 &dates[2]	dates[0] + 4
[1418] What is the equivalent array notation? int dates[10]; cout << (dates + 2) << endl; dates[2] + 2 &dates[2] dates[0] + 2 dates[2] dates[0] + 4	&dates[2]
[1419] What is the equivalent array notation? int dates[10]; cout << *(dates + 2) << endl; dates[2] + 2 dates[0] + 4 dates[2] &dates[2] dates[0] + 2	dates[2]
[1420] What is the equivalent array notation? int dates[10]; cout << (*dates) + 2 << endl; &dates[2] dates[0] + 2 dates[0] + 4 dates[2] dates[2] + 2	dates[0] + 2

int dates[10];
cout << *dates + 2 << endl;

&dates[2]
dates[2] + 2
dates[0] + 4
dates[2]
dates[0] + 2

[1422] What is the equivalent array notation?

int dates[10];
cout << *(dates + 2) + 2 << endl;

&dates[2]
dates[0] + 4
dates[0] + 2
dates[2]
dates[2] + 2

dates[2] + 2

[1423] What is the equivalent address-offset notation?

int a[] = {1, 2, 3, 4, 5, 6, 7};
int *p = a;

cout << a[1] * 2 << endl;

None of these
***p + 1** * 2
p + 1 * 2
(***p + 1)** * 2
***(p + 1)** * 2

***(p + 1)** * 2

[1424] What prints?

int a[] = {1, 3, 5, 7, 9};
int *p = a;
cout << *p++;
cout << *p << endl;

13
None of these
33
22
12

13

[1425] What prints?

int a[] = {1, 3, 5, 7, 9};
int *p = a;
cout << *++p;
cout << *p << endl;

33
13
None of these
22
12

33

int a[] = {1, 3, 5, 7, 9};
int *p = a;
cout << ++*p;
cout << *p << endl;

13
12
None of these
22
33

[1427] Which pointer initialization is illegal?

int a[] = {1, 3, 5, 7, 9};
int *p3 = &a[1];
None of these
int *p1 = a;
int *p4 = &a;
int *p2 = a + 3;

int *p4 = &a;

[1428] Which expression returns the number of countries?

string countries[] = {"Andorra", "Albania", . . . };

len(countries)
countries.length
sizeof(countries) * sizeof(countries[0])
sizeof(countries)
None of these

None of these

[1429] Which expression returns the number of countries?

string countries[] = {"Andorra", "Albania", . . . };

sizeof(countries)
len(countries)
sizeof(countries) / sizeof(string)
None of these
sizeof(countries) * sizeof(countries[0])

sizeof(countries) / sizeof(string)

[1430] Which expression returns the number of countries?

string countries[] = {"Andorra", "Albania", . . . };

len(countries)
sizeof(countries) * sizeof(countries[0])
sizeof(countries)
None of these
sizeof(countries) / sizeof(countries[0])

sizeof(countries) / sizeof(countries[0])

[1431] Which array definition is illegal?

int SIZE = 3;
int a1[SIZE];
int a2[3];
int a3[3]{};
int a4[] = {1, 2, 3};
int a5[3] = {1, 2};

a2
a3
None of these
a1
a5

a1

int SIZE = 3;
int a1[SIZE];
int a2[3];
int a3[3]{};
int a4[] = {1, 2, 3};
int a5[3] = {1, 2};

a3
a1
None of these
a5
a2

[1433] Which array definition is initialized to all zeros?

int SIZE = 3;
int a1[SIZE];
int a2[3];
int a3[3]{};
int a4[] = {1, 2, 3};
int a5[3] = {1, 2};

a5
a2
None of these
a3
a1

a3

[1434] Which array definition produces {0, 1, 2}?

int SIZE = 3;
int a1[SIZE];
int a2[3];
int a3[3]{};
int a4[] = {1, 2, 3};
int a5[3] = {1, 2};

a5
a3
None of these
a2
a1

None of these

[1435] Which array definition is illegal?

const int SIZE = 3;
int a1[SIZE];
int a2[3];
int a3[3]{};
int a4[] = {1, 2, 3};
int a5[2] = {1, 2, 3};
a2
a5
a3
None of these
a1

a5

```
int SIZE = 3;
int a1[SIZE];
int a2[3];
int a3[3]{};
int a4[] = {1, 2, 3};
int a5[3] = {1, 2};
```

a3
a5
a2
a1
None of these

In C++ using == to compare one array to another is permitted (if meaningless).

You must use an integral constant or literal to specify the size of a built-in C++ array.

The reinterpret_cast instruction changes way that a pointer's indirect value is interpreted.

If p is a pointer to a structure, and the structure contains a data member x, you can access the data member by using the notation: (*p).x

C++ arrays have no support for bound-checking.

In C++ assigning one array to another is illegal

The allocated size of a built-in C++ array cannot be changed during runtime.

The size of the array is not stored along with its elements.

If img is a pointer to the first byte in an image loaded into memory, Pixel is a structure as defined in your textbook, you can create a Pixel pointer pointing to the image by writing: Pixel **p = reinterpret_cast<Pixel** >(img);

The subscripts of a C++ array range from 0 to the array size - 1.

C++ arrays have no built-in functions for inserting and deleting.

A forward reference can be used when you want to use a pointer to a structure as a data member without first defining the entire structure.

The elements of a C++ array created in a function are allocated on the stack.

The elements of a C++ array created outside of a function are allocated in the static-storage area.

The elements of a C++ string array with no explicit initialization, created in a function will be set to the empty string.

Explicitly initializing an array like this: int a[3] = {1, 2, 3}; requires the size to be the same or larger than the number of elements supplied.

In C++ printing an array name prints the address of the first element in the array.

In C++ there is no separate array variable. The array name is a symbolic representation of the address of the first element in the array.

In C++ initializing an array with the contents of another is illegal.

C++ arrays produce undefined results if you access an element outside the array.

Explicitly initializing an array like this: int a[] = {1, 2, 3}; works in all versions of C++.

the same.

You may use any kind of integral variable to specify the size of a built-in C++ array.

The elements of a C++ string array with no explicit initialization, created in a function will be set to null.

Explicitly initializing an array like this: int a[3] = {1, 2, 3}; requires the size to be the same or smaller than the number of elements supplied.

In C++ using == to compare one array to another is illegal.

The allocated size of a built-in C++ array may be changed during runtime

If img is a pointer to the first byte in an image loaded into memory, Pixel is a structure as defined in your textbook, you can create a Pixel pointer pointing to the image by writing: Pixel **p = static_cast<Pixel** >(img);

The reinterpret_cast instruction produces a temporary value by converting its argument.

In C++ initializing an array with the contents of another is permitted.

C++ arrays use bound-checking when you access their elements with the at() member function.

The elements of a C++ array created in a function are allocated on the heap.

In C++ assigning one array to another is permitted.

C++ arrays throw an out_of_bounds exception if you access an element outside the array.

In C++ an array variable and the array elements are separate. The array variable contains the address of the first element in the array.

In C++ printing an array name prints the value of the first element in the array.

The elements of a C++ int array with no explicit initialization, created in a function will be set to zero.

C++ arrays can be allocated with a size of 0.

The static_cast instruction changes way that a pointer's indirect value is interpreted.

The size of the array is stored along with its elements.

The allocated size of a built-in C++ array may be changed during runtime

A forward reference can be used when you want to use a structure as a data member without first defining the entire structure.

The elements of a C++ array created outside of a function are allocated on the stack.

If p is a pointer to a structure, and the structure contains a data member x, you can access the data member by using the notation: *p->x

deleting.

Explicitly initializing an array like this: int a[] = {1, 2, 3}; only works in C++ 11.