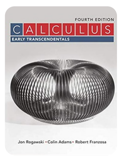
CH 10 QUIZ

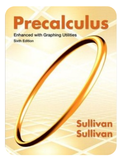
Share

9 studiers today ★ 4.5 (2 reviews)

Textbook solutions for this set



Calculus: Early Transcendentals
4th Edition • ISBN: 9781319050740
Colin Adams, Jon Rogawski, Robert Franzosa
8,982 solutions



Precalculus Enhanced with Graphing Utilities
6th Edition • ISBN: 9780321795465 (6 more)
Sullivan
9,377 solutions

Search for a textbook or question ›

Terms in this set (76)

What header file to you need to include to use the standard C++ error-handling classes?	<stdexcept>
The logic_error and runtime_error classes are defined in the header file ___.	stdexcept

```
What prints?

string s("hello");
try {
auto x = s.at(s.size());
cout << "one" << endl;
}
catch (const string& e) { cout << "two\n"; }
catch (exception& e) { cout << "three\n"; }
catch (...) { cout << "four\n"; }
```

three

```
What prints?

string s("hello");
try {
if (s.size() > 20) throw 42;
if (isupper(s.back())) throw "goodbye";
if (s == "Hello") throw string("hello");
s.at[s.size()] = 'x';
cout << "one\n";
}
catch (const int& e) { cout << "two\n"; }
catch (const string& e) { cout << "three\n"; }
catch (exception& e) { cout << "four\n"; }
catch (...) { cout << "five\n"; }
```

one

```
What prints?

string s("hello");
try {
if (s.size() > 2) throw s.size();
if (islower(s.back())) throw s.back();
if (s == "hello") throw string("hello");
s.at(s.size()) = 'x';
cout << "one\n";
}
catch (const int& e) { cout << "two\n"; }
catch (const string& e) { cout << "three\n"; }
catch (exception& e) { cout << "four\n"; }
catch (...) { cout << "five\n"; }

➢ I F (s.size() > 2) && throw s.size() && throw s.back()
```

five

```
What prints?

string s("hello");
try {
if (s.size() > 5) throw s.size();
if (isupper(s.back())) throw s.back();
if (s == "hello") throw string("hello");
s.at(s.size()) = 'x';
cout << "one\n";
}
catch (const string& e) { cout << "two\n"; }
catch (exception& e) { cout << "three\n"; }
catch (...) { cout << "four\n"; }
```

two

What prints?

```
string s("hello");
try {
if (s.size() > 2) throw 42; 🔆
if (islower(s.back())) throw "goodbye"; 🔆
if (s == "hello") throw string("hello");
s.at(s.size()) = 'x';
cout << "one\n";
}
catch (const int& e) { cout << "two\n"; }
catch (const string& e) { cout << "three\n"; }
catch (exception& e) { cout << "four\n"; }
catch (...) { cout << "five\n"; }
```

➢ I F (s.size() > 2) && throw 42; && throw "goodbye";

two

What prints?

```
string s("hello");
try {
if (s.size() > 20) throw 42; 🔆
if (islower(s.back())) throw "goodbye"; 🔆
if (s == "hello") throw string("hello");
s.at(s.size()) = 'x';
cout << "one\n";
}
catch (const int& e) { cout << "two\n"; }
catch (const string& e) { cout << "three\n"; }
catch (exception& e) { cout << "four\n"; }
catch (...) { cout << "five\n"; }
```

➢ I F (s.size() > 20) && throw 42; && (islower(s.back())) throw "goodbye";

five

What prints?

```
string s("hello");
try {
if (s.size() > 20) throw 42;
if (isupper(s.back())) throw "goodbye";
if (s == "Hello") throw string("hello");
s.at(s.size()) = 'x';
cout << "one\n";
}
catch (const int& e) { cout << "two\n"; }
catch (const string& e) { cout << "three\n"; }
catch (exception& e) { cout << "four\n"; }
catch (...) { cout << "five\n"; }
```

➢ I F (s.size() > 2) && throw 42; && (isupper(s.back())) throw "goodbye";

four

What is correct for # 1?

```
int main()
{
//1
{
string s = "hello";
cout << s.at(5) << endl;
}

// 2
// 3
( e)
{
cout << e. () << endl;
// 4
}

}
```

try

What is correct for # 2?

```
int main()
{
//1
{
string s = "hello";
cout << s.at(5) << endl;
}

// 2
// 3
( e)
{
cout << e. () << endl;
// 4
}
```

catch

What is correct for # 3?
 	
int main()
{
//1
{
string s = "hello";
cout << s.at(5) << endl;
}
 	
// 2
// 3
(e)
{
cout << e. () << endl;
// 4
}
 	
}
	exception&

What is correct for # 4?
int main()
{
//1
{
string s = "hello";
cout << s.at(5) << endl;
}
 	
// 2
// 3
(e)
{
cout << e. () << endl;
// 4
}
 	
}
	what

The C++11 standard library provides the function stoi() to convert a string to an integer. Which library is it found in?	string
What preprocessor directive is not used when you wish to create blocks of code that are only compiled under certain circumstances?	#define #ifdef #ifndef #if --> All of these may be used
Code that may cause an error should be placed in a _________ block and code that handles the error should be inside a _______ block?	try, catch
The class ___ is the base of the classes designed to handle exceptions	exception
A(n) ___ is an occurrence of an undesirable situation that can be detected during program execution	exception
What statement is used to signal other parts for your program that a particular error has occurred?	throw
The class ___ is designed to deal with illegal arguments used in a function call.	invalid_argument
What is the purpose of the throw statement?	It is used to pass control to an error handler when an error situation is detected.
The try block is followed by one or more ___ blocks.	catch

Which of the following blocks is designed to catch any type of exception?	catch(...){ }
The function ___ returns a string containing an appropriate message.	what
A catch block can have, at most, ___ catch block parameter(s).	one
What happens when this code fragment runs in C++ 11? cout << sqrt(-2) << endl;	sqrt() returns a not-a-number error value
Variables tested with the #if preprocessor directive are created using #define	True

A catch(...) will catch any kind of thrown exception	True
Functions with generic parameters are known as function templates.	True
A completion code is a special return value that means "the function failed to execute correctly."	True

Calling a function like to_string(3.5) is known as implicit instantiation	True
To use different versions of a function depending on the platform is called conditional compilation.	True
Building your code with more than one copy of a function leads to a clash of symbols.	True
A template function may be defined in a header file.	True
The predefined constant _cpluplus indicates which version of the C++ standard is being used	True
One of the main problems with the completion code strategy of error handling is that callers can ignore the return value without encountering any warnings	True
Calling a function like to_string<int>(3.5) is known as implicit instantiation.	False
The line: cin >> n; throws a runtime exception if n is an int and it tries to read the input "one".	False
The preprocessor operates on code after it has been compiled.	False
The directives #if defined(symbol) and #ifdef symbol mean, essentially, the same thing The directives #if defined(symbol) and #ifndef symbol mean, essentially, the same thing.	True False

A catch block may handle exception classes, as well as errors where int or string are thrown A catch block may only handle objects from classes derived from exception or logic_error A catch block specifies the type of exception it can catch and immediately terminates the program A catch block is a block of code where runtime or logical errors may occur	True False False False
You can report a logical error encountered in your code by using the throw keyword You can report a syntax error encountered in your code by using the throw keyword	True False
Functions with generic parameters may use the keyword class or the keyword typename for their type parameters Functions with generic parameters may use the keyword class or the keyword struct for their type parameters	True False
The #if preprocessor directive can compare integers The #if preprocessor directive may compare double literals but not variables	True False
The standard library version of sqrt(-2) returns the not-a-number error code The standard library version of sqrt(-2) throws a runtime exception because there is no possible answer	True False
You compiler or contains constants that can be used to identify the platform you are compiling on	True
A specialized error handling block of code, is called a catch block	True

The standard library version of stoi("UB-40") throws a runtime exception because there is no viable conversion The standard library version of stoi("UB-40") returns the not-a-number error code.	True False
The order of the catch blocks does not affect the program.	False
If no exception is thrown in a try block, all catch blocks associated with that try block are ignored.	True

When you throw an exception, control immediately jumps out of the current try block.	True
The preprocessor operates on code before it has been compiled.	False
The statement #if abs(-3) > 2 is legal.	False
A template function may be declared in a header file but must be defined in an implementation file.	False
The heading of a try block can contain ellipses in place of a parameter	False
When you throw an exception, control immediately returns from the current function	False
The line: ifstream in("x"); throws a runtime exception if a file x cannot be found	False
What happens when this code fragment runs? cout << stoi("12") << endl;	stoi() returns 12
What happens when this code fragment runs in C++ 11? cout << stoi("one") << endl;	It throws a runtime exception
Which of the following statements throws a valid exception in C++?	throw 2;
Suppose you have written a program that inputs data from a file. If the input file does not exist when the program executes, then you should choose which option?	Terminate the program.

What happens when this code fragment runs?
 	
istringstream in("12.5");
int n;
in >> n;
	n is set to 12

What happens when this code fragment runs?
 	
istringstream in("12");
int n;
in >> n;
	n is set to 12

What happens when this code fragment runs?
 	
istringstream in(".5");
int n;
in >> n;
	It sets an error state in in.

What happens when this code fragment runs in C++ 11?
 	
istringstream in("one");
int n;
in >> n;
	It sets an error state in in.

To deal with logical errors in a program, such as string subscript out of range or an invalid argument to a function call, several classes are derived from the class ___.	logic_error

Which line fails to work correctly?
 	
template <typename T>
void print(const T& item)
{
cout << item << endl;
}
	ANSWER --> None of these
	print(2 + 2);
	print(string("goodbye"));
	print(3 + 2.2);
	print("hello");

Assume s1 and s2 are C++ string objects. Which of these calls is illegal? template <typename T> void addem(T a, T b) {	addem(1.5, 2);

Which call below produces 5?
 	
template <typename T>
void addem(T a, T b)
{
cout << a << " + " << b << "->"
<< (a + b) << endl;
}
	addem<int>(3, 2.5);

Assume s1 and s2 are C++ string objects. Which of these calls is illegal?
 	
template <typename T>
void addem(T a, U b)
{
cout << a << " + " << b << "->"
<< (a + b) << endl;
}
	ANSWER --> None of these
	addem(1.5, 2);
	addem(s1, s2);
	addem(3, 4)
	addem(4.5, 5.5);

What happens when this code fragment compiles and runs? #define N #ifdef N cout << "Hello"; #else cout << "Goodbye"; #endif	prints "Hello"
What happens when this code fragment compiles and runs? #define N #ifndef N cout << "Hello"; #else cout << "Goodbye"; #endif	prints "Goodbye"

What term describes this block of code?
 	
#if _APPLE_
istringstream in(" .75");
int n = 3;
in >> n;
#endif
	conditional compilation

Complete the code fragment below, which is designed to throw an illegal_length exception if string variable accountNumber has more than seven characters.
 	
if (accountNumber.size() > 7)
{
_______________________;
}
	throw illegal_length("Account number exceeds maximum length");