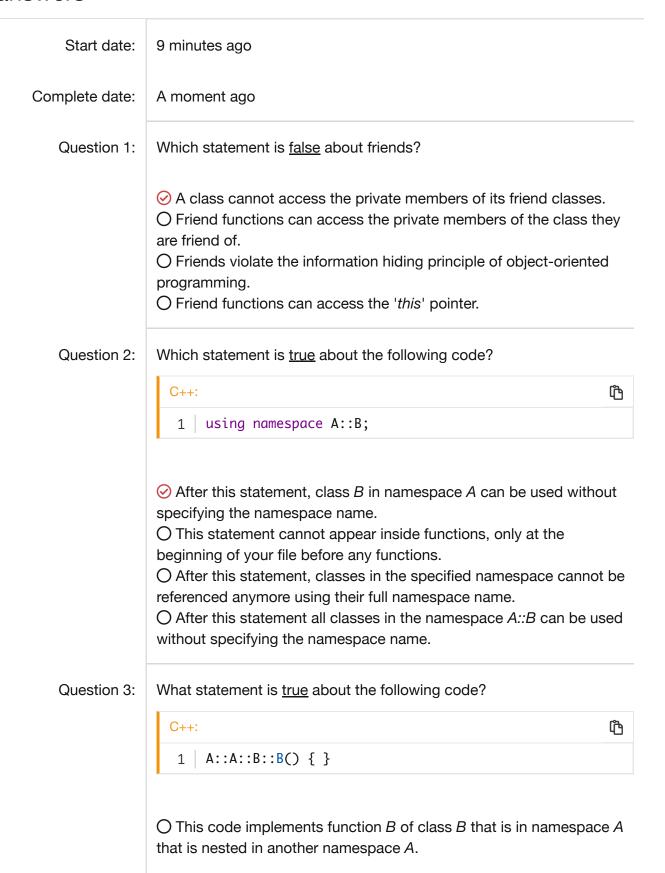
Quiz >

Review answers



- This code does not compile because you can't have two nested namespaces both called *A*.
- O This code does not compile because member functions in a namespace must be in a *namespace* {} block.
- O This code implements the default constructor of class *B* that is in namespace *A* that is nested in another namespace *A*.

Question 4:

Which of the options below is the best operator declaration to add and assign two objects of type *Complex* (*Complex*+=*Complex*)?

- O Complex& operator += (const Complex& c) const;
- O void operator += (const Complex& c);
- O Complex operator += (const Complex& c);

Question 5:

Which statement is <u>true</u> about the following code?

```
C++: 造
1 | delete[] x;
```

- O The code is wrong because the [] are missing the size to delete.
- O The code is wrong because the [] are not supported with delete.
- O It deallocates the first element of an array pointed by variable x.
- \bigcirc It deallocates an array pointed by variable x.

Question 6:

What is the output of the following program?

```
C++:
                                                                  ďЪ
     int x=10;
 1
 2
 3
     namespace
 4
 5
       int x=20;
 6
 7
     int main(int x, char* y□)
 8
 9
     {
       {
10
11
          int x=30;
          std::cout<<::x<<std::endl;</pre>
12
       }
13
14
       return 0;
     }
15
```

| /31/2020 | | Review answers: QUIZ 4 QuantNet Community |
|-----------------|-------------|--|
| | | ○ The number of arguments passed to the program.○ 10○ 20 |
| QuantNet | | z ⊠ ¢ 9 Q |
| | Question 7: | Which of the operator declarations below is the best way to support the index operator for integer indices ([int])? |
| | | ○ const Type& operator [] (int index) const; Type operator [] (int index); ○ const Type& operator [] (int index) const; Type& operator [] (int index); ○ Type& operator [] (int index) const; ○ Type operator [] (int index) const; |
| | Question 8: | What is the output of the following code? |
| | | C++: |
| | | <pre>1 int size=3; int* a=new int[size]; 2 for (int i=0; i<size; "<<(a+1)[0]<<",="" "<<*a+1<<="" "<<*a<<",="" 3="" a[i]="10-i;" i++)="" p="" std::cout<<a[1]<<",=""> 4 delete[] a;</size;></pre> |
| | | ○ 9, 10, 9, 9 ○ 9, [address of variable a], 9, [address of variable a + sizeof(int)] ○ 9, 10, 11, 9 ○ 9, 10, 9, 11 |
| | Question 9: | Which of the following statements is <u>true</u> about creating a copy constructor and assignment operator? |
| | | We need to create a copy constructor and assignment operator because the automatically generated copy constructor and assignment operator copy the data wrongly in certain situations. We do not need to create a copy constructor and assignment operator because the automatically generated copy constructor and assignment operator do already a member copy. We need to create a copy constructor and assignment operator because the automatically generated copy constructor and assignment operator do nothing. We need to create a copy constructor and assignment operator because the "canonical header file rules" dictates it. |

| Question 10: | Which statement is <u>false</u> about namespaces? |
|--------------|--|
| | ○ Namespaces can be nested. ○ A namespace must be compiled in its own .lib file. ○ Namespaces can prevent name collisions or be used to group functionality in logical blocks. ○ Multiple namespace blocks with the same name are possible. |
| Score: | 7 (70.00%) |
| Pass/Fail: | Failed |

Quiz >

Contact us Advertise Terms and rules Privacy policy Help Home ⋒

© QUANTNET INC