Quiz 02

Due May 16 at 11:59pm **Points** 5 **Questions** 5
Available until May 16 at 11:59pm **Time Limit** 15 Minutes

Instructions

Answer the following questions.

This quiz is no longer available as the course has been concluded.

Attempt History

	Attempt	Time	Score
LATEST	Attempt 1	6 minutes	5 out of 5

Score for this quiz: **5** out of 5
Submitted May 16 at 4:54pm
This attempt took 6 minutes.

Question 1

1 / 1 pts

Consider the following definition of class **Circle**.

```
class Circle {
   public:
      Circle();
      Circle(double radius);
      void setRadius(double radius);
      double getRadius() const;
   private:
      double radius;
};
```

This class has two constructors: **Circle()** and **Circle(double radius)**. Which one of the choices completes the second constructor, given below:

```
Circle::Circle(double radius) {

}
```

○ radius = r;

○ return radius;

○ radius = radius;

Correct!

● this -> radius = radius;

Since both the data field and input parameter are named the same (radius), you need to use "this" pointer to distinguish between them.

Question 2

1 / 1 pts

Consider the following definition of class **Circle** as the previous question.

```
class Circle {
   public:
      Circle();
      Circle(double radius);
      void setRadius(double radius);
      double getRadius() const;
   private:
      double radius;
};
```

How can you invoke the second constructor in function **main()** to create object **myCircle** with radius 3.9?

○ Circle(3.9) myCircle;

○ myCircle(3.9);

Correct!

● Circle myCircle(3.9);

Correct syntax.

○ Circle myCircle = setRadius(3.9);

Question 3

1 / 1 pts

Class **Circles** is defined below, which includes a private vector of **Circle** objects called **circleVector**.

```
class Circles {
   public:
      void readCircles();
      void printCircles() const;
   private:

};
```

What would be correct way to fill out the missing part?

○ vector<Circles> circleVector;

○ vector circleVector;

○ Circles circleVector;

Correct!

● vector<Circle> circleVector;

A vector of circles is vector<Circle>.

Question 4

1 / 1 pts

In class **Circles** defined in the previous question, **readCircles()** and **printCircles()** functions are defined as follows.

```
void Circles::readCircles(){
   double r;
   cout<<"Reading circle radiuses."<<endl<<"Input radius: ";
   cin>>r;
   while (r != 0.0) {
      Circle circle(r);

      cout<<"Input radius: ";
      cin>>r;
   }
}

void Circles::printCircles() const {
   Circle circle;
   for (int i = 0;                           i++) {
      circle =
      cout<<"Circle with radius "<<circle.getRadius()<<endl;
   }
}
```

What would be the correct way to fill out the missing parts?

Blue box: **circleVector.push_back(circle);**

Green box: **i < circleVector.length();**

○ Orange box: **circleVector[i];**

Blue box: **circleVector.add(circle);**

Green box: **i < circleVector.length();**

○ Orange box: **circleVector[i];**

Correct!

Blue box: **circleVector.push_back(circle);**

Green box: **i < circleVector.size();**

Orange box: **circleVector.at(i);**

●

Blue box: **circleVector.add(circle);**

Green box: **i < circleVector.size();**

○ Orange box: **circleVector.at(i);**

Question 5

1 / 1 pts

Consider class **Circle** and class **Circles**, as defined in the previous questions.

Let **circle.h** be the header file containing the **Circle** class definition including its data members and function signatures.

Let **circle.cpp** be the source file containing the definition of **Circle's** member functions.

Let **circles.h** be the header file containing the **Circles** class definition including its data member and function signatures.

Let **circles.cpp** be the source file containing the definition of **Circles'** member functions.

Which files must be included in the definition of these files?

In circle.h: **#include "circle.cpp"**

○ In circles.cpp: **#include "circles.h"**

In circle.h: **#include "circle.cpp"**

○ In circles.h: **#include "circles.cpp"** and **#include "circle.h"**

Correct!

In circles.h: **#include "circle.h"**

In circle.cpp: **#include "circle.h"**

● In circles.cpp: **#include "circles.h"**

In circle.cpp: **#include "circle.h"**

In circles.h: **#include "circles.cpp"**

○ In circle.h: **#include "circles.h"**

Quiz Score: **5** out of 5