

Chapter 8

Function Templates Exercises

Module 2: High-Level Programming II

Copyright Notice

Copyright © 2016 DigiPen (USA) Corp. and its owners. All Rights Reserved

No parts of this publication may be copied or distributed, transmitted, transcribed, stored in a retrieval system, or translated into any human or computer language without the express written permission of DigiPen (USA) Corp., 9931 Willows Road NE, Redmond, WA 98052

Trademarks

DigiPen® is a registered trademark of DigiPen (USA) Corp.

All other product names mentioned in this booklet are trademarks or registered trademarks of their respective companies and are hereby acknowledged.

1. Implement a templated **Add** function that works with the following code and results with the below output.

PS: Both parameters will have the same type

Code	<pre>int main(void) { int resultInt, i1 = 5, i2 = 10; float resultFloat, f1 = 12.2f, f2 = 5.4f; resultInt = Add(i1, i2); std::cout << "resultInt = " << resultInt << std::endl; resultFloat = Add(f1, f2); std::cout << "resultFloat = " << resultFloat << std::endl; return 0; }</pre>
Output	<pre>resultInt = 15 resultFloat = 17.6</pre>
Answer	

2. Implement a templated **Add** function that works with the following code and results with the below output.

PS: parameters don't have to be of the same type

Code	<pre> int main(void) { int resultInt, i1 = 5, i2 = 10; float resultFloat, f1 = 12.2f, f2 = 5.4f; double resultDouble, d = 5.7; resultInt = Add<int>(i1, i2); std::cout << "resultInt = " << resultInt << std::endl; resultFloat = Add<float>(f1, f2); std::cout << "resultFloat = " << resultFloat << std::endl; resultDouble = Add<double>(i1, d); std::cout << "resultDouble = " << resultDouble << std::endl; resultInt = Add<int, float, int>(f1, i1); //will cause a warning std::cout << "resultInt = " << resultInt << std::endl; return 0; } </pre>
Output	<pre> resultInt = 15 resultFloat = 17.6 resultDouble = 10.7 resultInt = 17 </pre>
Answer	

3. Implement a templated **Average** function that works with the following code and results with the below output.

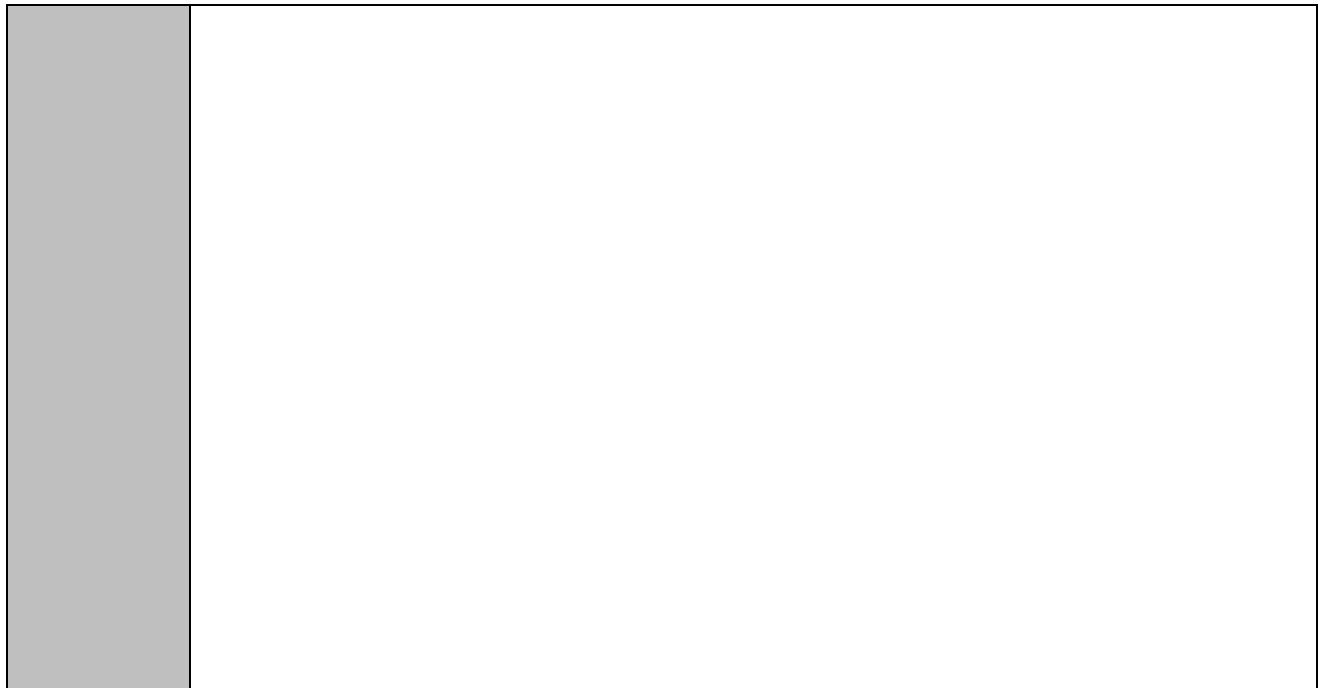
PS: The return type has to be a double. Make sure no warnings are generated.

Code	<pre>int main(void) { int i1 = 5, i2 = 10; float f = 12.2f; double result, d = 5.7; result = Average(i1, i2); std::cout << "result = " << result << std::endl; result = Average(i1, f); std::cout << "result = " << result << std::endl; result = Average(d, f); std::cout << "result = " << result << std::endl; return 0; }</pre>
Output	<pre>result = 7.5 result = 8.6 result = 8.95</pre>
Answer	

4. Implement the required templated **Average** function(s) that works with the following code and results with the below output.

PS: The return type has to be a double. Make sure no warnings are generated.

Code	<pre> int main(void) { int i1 = 5, i2 = 10; float f = 12.2f; double result, d = 5.7; int num1[] = { 1, 2, 3, 4, 5, 6, 7, 8, 9, 10 }; float num2[] = { 1.4f, 2.3f, 4.3f, 6.0f}; result = Average(i1, i2); std::cout << "result = " << result << std::endl; result = Average(i1, f); std::cout << "result = " << result << std::endl; result = Average(d, f); std::cout << "result = " << result << std::endl; result = Average(num1, 10); std::cout << "result = " << result << std::endl; result = Average(num2, 4); std::cout << "result = " << result << std::endl; return 0; } </pre>
Output	<pre> result = 7.5 result = 8.6 result = 8.95 result = 5.5 result = 3.5 </pre>
Answer	



5. Implement the required templated **Compare** function(s) that works with the following code and results with the below output.

PS: The return type has to be an int.

- **-1 is returned if the first parameter is smaller than the second**
- **1 is returned if the first parameter is greater than the second**
- **0 is returned if both parameters are equal**

Make sure no warnings are generated.

Code

```
int main(void)
{
    int result = 0;
    int i1 = 5, i2 = 10, i3 = 5;
    float f1 = 12.2f, f2 = 13.5f;

    const char sentence1[] = "Hello";
    const char sentence2[] = "World";
    const char sentence3[] = "Hello";

    std::cout << my_compare(i1, i2) << std::endl;
    std::cout << my_compare(i2, i3) << std::endl;
    std::cout << my_compare(i1, i3) << std::endl;
    std::cout << my_compare(f1, f2) << std::endl;

    std::cout << my_compare(sentence1, sentence2) << std::endl;
    std::cout << my_compare(sentence1, sentence3) << std::endl;

    return 0;
}
```

Output	-1 1 0 -1 -1 0
Answer	

6. Given the following functions

1	<pre>template <typename T> T square(T value) { return value * value; }</pre>
2	<pre>template <> int square<int>(int value) { return value * value; }</pre>
3	<pre>int square(int value) { return value * value; }</pre>

which function is called by the following statements?

Statement	Number of function called
square(2);	
square<int>(2);	
square(2.0f);	
square<double>(2.3);	