Review: TEMPLATES

1. Why is the advantage of using a function template instead of overloaded functions?

Function templates allow programmers to write a single function definition that works with different types instead of having to write the same function definition for each type.

2. Find the errors, if any.

```
A.                       // OK
template <class T> T fun(T num)
{
    return 10 * num;
};

B.
template <class T>
T fun(T num)
{
    return 10 * T;       // T represents a type
};

C.                       // OK
template <class T>
T fun(T num)
{
    return 10 * num;
};

D.
template <class T, class U>
T fun(T num1, T num2)    // U - not used
{
    return num1 + num2;
};

E.
template <class T>
T fun( T, T);

int main ()
{
    int    a = 10;
    double b = 20.5, c;

    c = fun(a, b);       // AMBIGUOUS! It should be:
                         // c = fun<double>(a, b);
    cout << c << endl;
    return 0;
}
/// ***************************************
template <class T>
T fun(T num1, T num2)
{
    return (num1 + num2);
}
```

Review: TEMPLATES

3. Make the printReverse() function a template.

```
#include <iostream>
using namespace std;

template <class T>
void printReverse(T [], int);

// void printReverse(double list[], int n);
// void printReverse(int list[], int n);

int main( void )
{
    int n = 6;
    double listA[10] = {12.1, 30.5, 14.2, 70.5, 32.0, 90.2};
    int    listB[10] = {12, 30, 14, 70, 32, 90};

    printReverse<double>(listA, n);
    printReverse<int>(listB, n);

    return 0;
}
/**
   This function takes an array and its actual size
   and it prints it in reverse order
*/
template <class T>
void printReverse(T list[], int n)
{
    for (int i = n - 1; i >= 0; i--)
    {
        cout << list[i] << " ";
    }
    cout << endl;
}
/** This function takes an array of doubles and its actual size
    and it prints it in reverse order
void printReverse(double list[], int n)
{
    for (int i = n - 1; i >= 0; i--)
    {
        cout << list[i] << " ";
    }
    cout << endl;
}
*/
/** This function takes an array of ints and its actual size
    and it prints it in reverse order
void printReverse(int list[], int n)
{
    for (int i = n - 1; i >= 0; i--)
    {
        cout << list[i] << " ";
    }
    cout << endl;
}
*/
```