Code Test

This code test is designed to mimic a code test you would encounter when applying for a job. Participating in the code test is designed to give you feedback on your own level of knowledge and understanding of games programming as a whole. If you find a particular question challenging or spend a lot of time on it, then this is perhaps an area you need to develop more. The code test will not be graded or marked.

This test should only take you an hour to complete.

Question 1

There are several compilation errors in the code at the following link. Fix as many of them as you can. Make a note of the error and the solution.

```
Foo( int x )
{
  int& a;
  int  returnValue = 0;

  for( int i = 0; i < x i++ )
  {
       returnValue += i;
  }
}

int main()
{
  int num;
  num = Foo( 10 );
  return 0;
}
```

Question 2

There are (at least) 5 ways in which the following code could either go wrong or have unexpected side effects at **runtime**. Please list as many as you can.

```
int Foo( myClass* pObject, int param )
{
    float x;
    float y;
    for( uint i = 10; i > param ; i-- )
    {
              y = pObject->myFloat + x * i;
    }
    return y;
}
```

Question 3

Write an application that measures the weight of a person's wallet using the following information. You are free to use or create any data structures you feel are appropriate.

Below is some data that represents coins and their corresponding weight.

- 50 cent - 0.01g
- 1 euro - 0.1g
- 2 euro - 1.0g

Write a single program that answers all of the questions below:

a) Firstly write a class to hold the data for a single Coin.

b) Now declare a static array of that class which holds all the data above.

c) Wallets can also contain notes. Now add an array of a class to hold the data below:

- 5 euro - 0.001g
- 10 euro - 0.005g
- 100 euro - 0.009g

d) A person has a wallet, and in that wallet they have money. Below is some data:

Tom's Wallet

- 50 cent
- 10 euro
- 100 euro

Add a class to hold this information and declare an instance of that class to represent Tom’s Wallet data.

f) Now add a function which calculates the weight of a person's wallet and write a running application that calculates the weight of Tom's wallet.

h) There is another currency with the data below (a subset of Sterling):

- 50 penny - 0.01g
- 1 pound - 0.1g
- 5 pounds - 0.001g
- 10 pounds - 0.005g

Jack has a wallet which contains both sets of currency

Jack's wallet

- 50 penny
- 10 euro
- 5 pounds

Write a running application that calculates the total weight of Tom and Jack's wallet.

Question 4

Evaluate the following logic statements and fill in TRUE or FALSE:

a) 1 || 1 || 0 =

b) !((1 && 0) || !1) =

c) !((2 || 1) || ! (1 || 0)) =

d) (A||B) && !(A||B) =

e) (A&&B) || (!A&&!B) =

Question 5

Given the following Vector3 class, write the implementation for the three functions. You may use basic trigonometry functions by #including <math.h>. You may also use sqrtf(). You should avoid using functions that involve double precision types. You should attempt to minimise the use of expensive arithmetic operations.

a) GetMagnitude() should return a float representing the magnitude of the vector.
b) Normalise() should modify the vector so that is has unit length.
c) GetAngleBetween() should return a float representing the angle between the two input vectors.
d) GetPerpVector() should return a third vector which is perpendicular to the two input vectors. Some credit will be given for mentioning the name of the operation even if you cannot remember or derive the formula.
e) What does the use of 'const' mean in the declaration of GetAngleBetween()?

//

```
class Vector3
{
public:
        Vector3()
        {}

        Vector3( float fX, float fY, float fZ )
        : m_fX(fX)
        , m_fY(fY)
        , m_fZ(fZ)
        {}

        float GetMagnitude() const;
        void Normalise();
        static float GetAngleBetween(const Vector3& xVec1, const Vector3& xVec2);
        static Vector3 GetPerpVector(const Vector3& xVec1, const Vector3& xVec2);

private:
        float m_fX;
        float m_fY;
        float m_fZ;
};
```

// Part (a) - write your implementation here:

```
float Vector3::GetMagnitude() const
{
	return 0;
}
```

// Part (b) - write your implementation here:

```
void Vector3::Normalise()
{

}
```

// Part (c) - write your implementation here:

```
float Vector3::GetAngleBetween(const Vector3& xVec1, const Vector3& xVec2)
{
	return 0;
}
```

// Part (d) - write your implementation here:

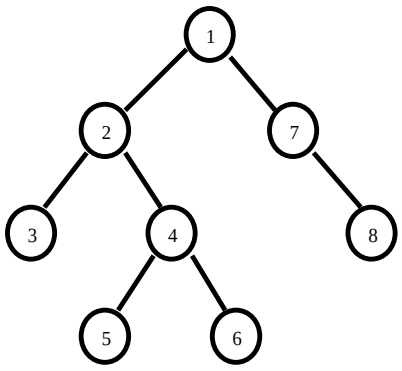
```
Vector3 Vector3::GetPerpVector(const Vector3& xVec1, const Vector3& xVec2)
{
	Vector3 xResult;
	return xResult;
}
```

Question 6

A tree data structure composed of instances of a T_Node data structure. This structure contains a value (m_number) and pointers to other nodes (m_parent, m_left, m_right). If a node does not have a lift, right or parent node these pointers will be NULL, otherwise it will contain a valid pointer.

```
struct T_Node
{
    int     m_number;

    T_Node*m_parent;
    T_Node*m_left;
    T_Node*m_right;
};
```



You are to create a function to **recursively** visit each node in the tree and output the value of m_number. This routine should visit the nodes in the numerical order shown above, thus outputting "12345678".