CS 112 – Computer Science Fundamentals 2
Final Exam – Reference Notes

Code and descriptions for the Artwork and Sculpture classes

```
class Artwork {
    public:
        Artwork(string aTitle, double aPrice);
        Artwork();

        string getTitle() const;
        double getPrice() const;

        void setTitle(string newTitle);
        void setPrice(double newPrice);

        string toString() const; // Place all data member values in a string

    private:
        string title;
        double price;
};
```

- **Sculpture** is a derived class, derived from base class **Artwork**. In addition to what it inherits from **Artwork**, a **Sculpture** instance should also have:
  - An appropriate data field for the **material** (stone, clay, wood, etc.) from which it is made
  - An appropriate **accessor** method and an appropriate **mutator** method for the above data member
  - An explicitly defined **no-argument constructor**
  - An explicitly defined **constructor** allowing the user to specify values for **all** of the data members (the data members from the base class and those specific to a **Sculpture**)
  - An appropriate redefined version of method **toString**

A version of ArtNode.h for implementing a linked list of Artwork objects

```
#ifndef ARTNODE_H
#define ARTNODE_H

#include <string>
#include "Artwork.h"
using namespace std;

class ArtNode {
    public:
        ArtNode();
        ArtNode(Artwork initArt, ArtNode *initNext);

        Artwork  getArt() const;
        ArtNode *getNext() const;

        void setArt(Artwork newArt);
        void setNext(ArtNode *newNext);

    private:
        Artwork  art;
        ArtNode *next;

};

#endif
```

A version of ArtNode.cpp

```
#include <cstdlib>
#include "Artwork.h"
#include "ArtNode.h"
using namespace std;

ArtNode::ArtNode() {
    Artwork defaultArtwork;
    art = defaultArtwork;
    next = NULL;
}

ArtNode::ArtNode(Artwork initArt, ArtNode *initNext) {
    art = initArt;
    next = initNext;
}

Artwork  ArtNode::getArt() const {
    return art;
}

ArtNode *ArtNode::getNext() const {
    return next;
}

void ArtNode::setArt(Artwork newArt) {
    art = newArt;
}

void ArtNode::setNext(ArtNode *newNext) {
    next = newNext;
}
```

Template for a main function

```
#include <cstdlib>
#include <iostream>
#include <string>
#include <cmath>
// #include "called_below.h"
using namespace std;

int main() {
    cout << boolalpha;
    // Insert code here
    return EXIT_SUCCESS;
}
```

Template for a non-main function

```
#include <cstdlib>
#include <iostream>
#include <string>
#include <cmath>
// #include "something.h"
using namespace std;

return_type function_name(type param1, …) {
    // Insert code here
}
```

Template for a .h file

```
#IFNDEF X_H    // replace X with the NAME of your function or class in all-caps
#DEFINE X_H

// #include lines as needed for this function's arguments or
// named constant declarations – for example, it would need:
#include <string>
// if function has any string parameters, for example.

using namespace std;

// any named constant declarations

// header(s) of function(s) or method(s) in x

#ENDIF  /* X_H */
```