Exercise 1 – Item Class - Constructors and Destructors

a) In the Item.h header file
   1. Write the prototype of the default constructor
   2. Write the prototype of a overloaded constructor (see main program to identify the parameters)
   3. Write the prototype of the destructor

b) In Item.cpp program
   4. Implement the default constructor (initialize all properties to zero)
   5. Implement the overloaded constructor
   6. Implement the destructor (You should print "Destructor Called")

Run the program you should get the following output if your program is correct

```
Item : 0
Discounted Price 0
Item : 100
Discounted Price 800
Destructor Called
Destructor Called
```

Item.cpp	Item.h

#include "Item.h"
#include <iostream>
using namespace std;
 	
// 4. Default Constructor Implementation
 	
// 5. Overloaded Constructor Implementation
 	
// 6. Implement Destructor (display "Destructor Called")
 	
// DO NOT CHANGE THE CODE GIVEN BELOW
void Item::setDiscount(float pdiscount) {
  discount = pdiscount;
}
 	
float Item::getDiscount() {
  return discount;
}
 	
float Item::discountedPrice() {
  return unitPrice - unitPrice * discount/100;
	// ONLY WRITE THE PROTOTYPES OF THE TWO CONSTRUCTORS
	// AND THE DESTRUCTOR
	
	class Item {
	  private:
	  int itemCode;
	  float unitPrice;
	  float discount; // out of 100 e.g. discount = 15
	  public:
	  // 1. Default Constructor
	  // 2. Overloaded Constructor
	  // 3. Destructor
	
	  void setDiscount(float punitPrice);
	  float getDiscount();
	  float discountedPrice();
	  void display();
	};

```
}

void Item::display() {
  cout << "Item : " << itemCode << endl;
  cout << "Discounted Price " << discountedPrice()
<< endl;
}
// END DO NOT CHANGE ABOVE CODE
```

main.cpp

```
#include <iostream>
#include "Item.h"
using namespace std;

// DO NOT CHANGE THIS CODE
int main() {
   Item myItem;
   myItem.display();
   Item myItem2(100, 1000);
   myItem2.setDiscount(20);
   myItem2.display();

   return 0;
}

// DO NOT CHANGE ABOVE CODE
```

Exercise 2 – Shape Classes - Dynamic Objects

- The Rectangle class is implemented in Rectangle.h and Rectangle.cpp
- The Circle class is implemented in Circle.h and Circle.cpp
- You should not change the code in the Rectangle and Circle classe
- In the man.cpp program create two dynamic objects as instructed below.

1. Create a dynamic Rectangle type variable (pointer)
2. Create a dynamic Rectangle Object set the length and width that was input from the keyboard
3. Create a dynamic Circle type variable (pointer)
4. Create a dynamic Circle Object set radius that was input from the keyboard
5. call the display method of the Rectangle Object
6. delete the Rectangle Object from memory
7. delete the Circle Object from memory

Do not change any other code in the main.cpp

Circle.cpp

```
#include "Circle.h"
#include <iostream>
using namespace std;

// Default Constructor Implementation
Circle::Circle() {
   radius = 0;
}
// Overloaded Constructor Implementation
Circle::Circle(int r) {
  radius = r;
}
// Destructor Implementation
Circle::~Circle() {
  cout << "Circle Destructor called" << endl;
}
void Circle::display() {
   cout << "Circle Area = " << calcArea() << endl;
}

float Circle::calcArea() {
   return 22.0/7*radius*radius;
}
```

Circle.h

```
class Circle {
  private:
    int radius;
  public:
    Circle(); // Default Constructor
    Circle(int r); // Overloaded Constructor
    ~Circle(); // Destructor
    void display();
    float calcArea();
};
```

Rectangle.cpp

```
#include "Rectangle.h"
#include <iostream>
using namespace std;

// Default Constructor Implementation
Rectangle::Rectangle() {
  length = 0;
  width = 0;
}

// Overloaded Constructor Implemenation
Rectangle::Rectangle(int l, int w) {
 length = l;
 width = w;
}
// Destructor Implementation
Rectangle::~Rectangle() {
 cout << "Rectangle Destructor called" << endl;
}

void Rectangle::display() {
  cout << "Rectangle Area = " << calcArea() <<
endl;
}

int Rectangle::calcArea() {
 return length * width;
}
```

Rectangle.h

```
class Rectangle {
  private:
    int length;
    int width;
  public:
    Rectangle(); // Default Constructor
    Rectangle(int l, int w); // Overloaded
Constructor
    ~Rectangle(); // Destructor
    int calcArea();
    void display();
};
```

main.cpp

```
#include "Rectangle.h"
#include "Circle.h"
#include <iostream>
using namespace std;

int main() {
 // ========= DO NOT CHANGE THE INPUT CODE BELOW
==================
  int length, width, radius;
  cout << "Enter length of Rectangle : ";
  cin >> length;
  cout << "Enter width of Rectangle : ";
  cin >> width;
```

```
  cout << "Enter radius of Circle : ";
  cin >> radius;
  // ======== DO NOT CHANGE THE CODE GIVEN ABOVE
====================

  // 1. Create a dynamic Rectangle type variable (pointer)
  // 2. Create a dynamic Rectangle Object set the length and width that was
input from the keyboard
  // 3. Create a dynamic Circle type variable (pointer)
  // 4. Create a dynamic Circle Object set radius that was input from the
keyboard
  // 5. call the display method of the Rectangle Object
  // 6. call the display method of the Circle Object
  // 7. delete the Rectangle Object from memory
  // 8. delete the Circle Object from memory


  // ============ DO NOT CHANGE THE CODE BELOW
====================
  cout << "End of Program" << endl;

}
```