Revision Part 2
Learning Objectives

- Basic Structures of C++
- Classes and inheritance
- Arrays
- Dynamic Memory

Part 1: Basics Structures of C++

1.Write out some basic codes to get input from the users

```
#include<iostream.h>
Using namespace std;
void main()
{

  char ch;
  cout<<"Press y to see greeting: ";
  cin>>ch;
  if(ch=='y' || ch=='Y')
  {
    cout<<"\n Welcome Guest";
  }
 return 0;
}
```

2. What would the output be if the user enters 10 ?

```
#include <iostream>
using namespace std;
void func1()
{
cout << "Able was I\n";
}
void func2()
{
cout << "I saw Elba\n";
}
int main()
{
int input;
cout << "Enter a number: ";
cin >> input;
if (input < 10)
{
func1();
func2();
}
else
{
func2();
func1();
}
```

return 0;
}

3. Write out a simple array and add up all the integer values in an array
4. Create a sample structure that can represent an employee.

5. Write out some cin and cout commands that can enter the details of 5 employees using an array called employee.
6. Write a program to print the value of the address of the pointer to a variable whose value is input from user.

7. Provide a struct that can keep values for a product.

8. Write the declaration for a variable called **itemPtr** that can hold the address of an Item struct. Then dynamically allocate an Item struct and set it's item number to 10, it's description to "Granny Smith Apples", and it's price to 1.89.

9. Write a function that takes as parameters a pointer to an Item struct, an item number, a description, and a price. Your function should assign the item number, description, and price to the struct data members.

10. Write a declaration for an array of 150 pointers to structs of type Item called **itemPtrs**. Then write code to dynamically allocate an **Item** struct, store it's address in the first element of **itemPtrs,** then set it's item number to 10, it's description to "Granny Smith Apples", and it's price to 1.89.

Part 2: Classes and inheritance

1. Explain the basic concept of inheritance with some basic codes
   In C++, it is possible to inherit attributes and methods from one class to another. We group the "inheritance concept" into two categories:

   derived class (child) - the class that inherits from another class
   base class (parent) - the class being inherited from
   To inherit from a class, use the : symbol.

   In the example below, the Car class (child) inherits the attributes and methods from the Vehicle class (parent):

```
// Base class
class Vehicle {
  public:
    string brand = "Ford";
    void honk() {
      cout << "Tuut, tuut! \n" ;
    }
};

// Derived class
class Car: public Vehicle {
  public:
    string model = "Mustang";
```

```
};

int main() {
  Car myCar;
  myCar.honk();
  cout << myCar.brand + " " + myCar.model;
  return 0;
}
```