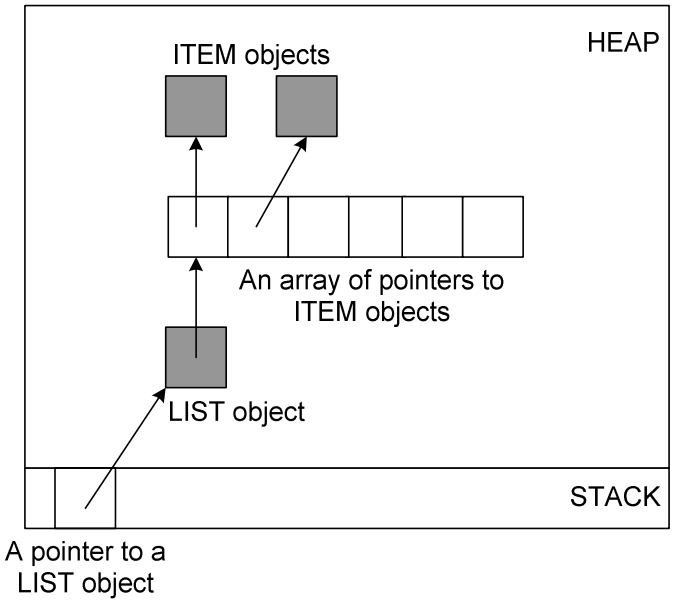
Student Name: _____ Student ID: _____

ENG236 Computer Programming
Solution to Programming Quiz 13 (22 November 2009)

Instructions: open book, deadline: 12:30pm according to WebCT

A LIST object consists of a number ITEM objects. An implementation of the LIST object is to use an array of pointers created in the heap and each pointer is used to point to an ITEM object which is also stored in the heap. Moreover, we may create a LIST object in the heap. The figure below depicts the relationship among the LIST object, ITEM objects, the array of pointers, and the pointer to a LIST object.



Consider the class ITEM and class LIST below:

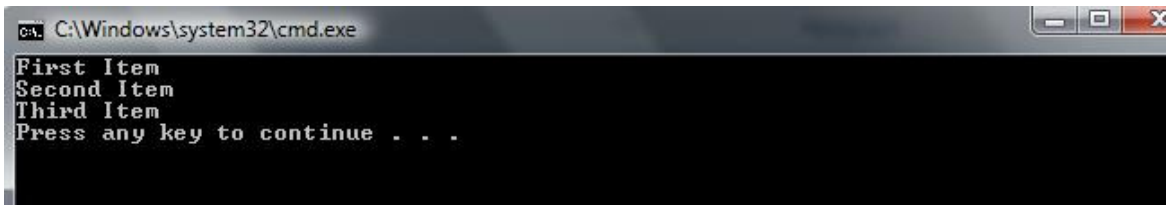
```
// A class for items put on a list
class ITEM
{
public:
      // constructor
      ITEM(char name[]) {strcpy_s(itemName, name);};

      // set the name of the item
      void setName(char name[]) {strcpy_s(itemName, name);};

      // return the name of the item
      char *getName() {return itemName;};

      // print the name of the item
      void printItemName() const {cout << itemName << endl;};
private:
      // the item's name
      char itemName[80];
};
```

Student Name: _____ Student ID: _____

```
// A class for lists
class LIST
{
public:
   // A constructor to set numberOfItems to 0 and to create in the heap
   // an array of 100 pointers to ITEM objects. The pointer to the
   // array is stored in pArray.
   LIST();

   // A destructor to remove all the heap memories occupied by a LIST
   // object and all the ITEM objects pointed to by the LIST object.
   ~LIST();

   // Add an ITEM object to the next available slot (in the array of) a
   // LIST object. The first item is stored in the first element of the
   // array. The ITEM object must be created first and being stored
   // in a LIST object.
   void AddItem(ITEM * newItem);

   // Print out the names of all ITEM objects stored in a LIST object.
   // Each name is printed on a new line.
   void PrintList();

   // Get the number of ITEM objects stored in a LIST object.
   int GetnumberOfItems() const;

private:
      // the number of ITEM objects stored in the LIST
      int numberOfItems;

      // a pointer pointing to the first element of the array created
      // in the heap.
      ITEM **pArray;
};
```

There are two tasks for you:

1. (200 marks) Implement the public function for the class LIST.
2. (50 marks) Implement a main() function that will create a LIST object of three ITEM objects. The names of the three items are “First Item”, “Second Item”, and “Third Item”. Therefore, the output should look like the one below. Note that you must delete all the heap memories created by this program.



Solution:

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

// A class for items put on a list
class ITEM
{
public:
      // constructor
      ITEM(char name[]) {strcpy_s(itemName, name);};

      // set the name of the item
      void setName(char name[]) {strcpy_s(itemName, name);};

      // return the name of the item
      char *getName() {return itemName;};

      // print the name of the item
      void printItemName() const {cout << itemName << endl;};
private:
      // the item's name
      char itemName[80];
};

// A class for lists
class LIST
{
public:
   // A constructor to set numberOfItems to 0 and to create in the heap
   // an array of 100 pointers to ITEM objects. The pointer to the
   // array is stored in pArray.
   LIST();

   // A destructor to remove all the heap memories occupied by a LIST
   // object and all the ITEM objects pointed to by the LIST object.
   ~LIST();

   // Add an ITEM object to the next available slot (in the array of) a
   // LIST object. The first item is stored in the first element of the
   // array. The ITEM object must be created first and being stored
   // in a LIST object.
   void AddItem(ITEM * newItem);

   // Print out the names of all ITEM objects stored in a LIST object.
   // Each name is printed on a new line.
   void PrintList();

   // Get the number of ITEM objects stored in a LIST object.
   int GetnumberOfItems() const;

private:
      // the number of ITEM objects stored in the LIST
      int numberOfItems;

      // a pointer pointing to the first element of the array created
```

```
      // in the heap.
      ITEM **pArray;
};

int main()
{
   LIST *pList = new LIST;

   pList->AddItem(new ITEM("First Item"));
   pList->AddItem(new ITEM("Second Item"));
   pList->AddItem(new ITEM("Third Item"));
   pList->PrintList();

   delete pList;

   return 0;
}


LIST::LIST()
{
    numberOfItems = 0;
    pArray = new ITEM *[100];
}

LIST::~LIST()
{
    for (int i=0; i<GetnumberOfItems(); i++)
        delete pArray[i];
    delete [] pArray;
}

void LIST::AddItem(ITEM *newItem)
{
    pArray[numberOfItems]=newItem;
    numberOfItems++;
}

void LIST::PrintList()
{
    for (int i=0; i<GetnumberOfItems(); i++)
        pArray[i]->printItemName();
}

int LIST::GetnumberOfItems() const
{
    return numberOfItems;
}
```