THE HONG KONG POLYTECHNIC UNIVERSITY
Department of Electronic and Information Engineering

Computer Programming (ENG236) - Homework 4

PART 1

By using Visual Studio .NET, implement a console application that creates a linked list of 4 objects of the class CAT. Ask the user to give each CAT object a name. The class CAT is given below:

```
class CAT
{
public:
      CAT() {pNext = 0;}            // Constructor – set $pNext to NULL by default
      void setName(char *nm)
      {                             // copy nm to $name and make sure there must
            strncpy (name,nm,79);   // be a NULL character at the end of name
            name[79] = NULL;
      }
      char * getName()              // get the name of a CAT
      {
            return name;
      }
      void setNext(CAT *pC)         // $pNext = pC
      {
            pNext = pC;
      }
      CAT * getNext()               // return $pNext
      {
            return pNext;
      }
private:
      char name[80];                // Keep the name of the CAT
      CAT * pNext;
};
```

After the linked list has been created, ask the user to give an arbitrary CAT name. If the name the user entered can be found in the linked list, show the message "`xxx is found!`", where `xxx` is the name of the CAT the user entered. Otherwise, return the message "`xxx cannot be found!`".

Finally, free the memory used for storing the linked list before quitting the program.

PART 2

Redesign the program you have written in Part 1 by following the guidelines below:

1. Show the following menu on the screen in Console mode:

```
A. Insert a CAT to the linked list
B. Delete a CAT from the linked list
C. Show all CAT objects
Q. Quit

Your choice (A, B, C or Q):
```

2. Your program should repeatedly ask the user to input their choice.

   a. If the user chooses A, let the user enter the name of a CAT and the position of the linked list where this CAT object is to be inserted. The first item is in position 0. Report an error message if the

position cannot be realized in practice. Get back to the main menu after adding the `CAT` object in the linked list.

b. If the user chooses `B`, let the user enter the position of the linked list where the `CAT` object is to be deleted. Report an error message if the position cannot be realized in practice. Get back to the main menu after deleting the `CAT` object in the linked list.

c. If the user chooses `C`, the name(s) of all `CAT` object(s) in the linked list should be shown.

d. If user chooses `Q`, show the message "`Goodbye!`" and quit the program. Remember to free all memory that you used to store the `CAT` objects in the linked list.

Notes:

i. Develop your program using C++ under the Visual Studio 2005 environment.
ii. It is mandatory to use the linked list approach to implement the program.
iii. Remember to check whether the insert or delete position is valid.
iv. You need to follow exactly the requirements of the questions as shown above when developing your programs.
v. Try to explain your program as clear as possible using comments.

```
//Part 1
#include <iostream>
using namespace std;

class CAT
{
public:
	CAT() {pNext = 0;}			// Constructor - set $pNext to NULL by default
	void setName(char *nm)
	{					// copy nm to $name and make sure there must
		strncpy (name,nm,79);		// be a NULL character at the end of name
		name[79] = NULL;
	}
	char * getName()			// get the name of a CAT
	{
		return name;
	}
	void setNext(CAT *pC)			// $pNext = pC
	{
		pNext = pC;
	}
	CAT * getNext()				// return $pNext
	{
		return pNext;
	}
private:
	char name[80];				// Keep the name of the CAT
	CAT * pNext;
};

CAT * create(int);			// Function prototype

int main()
{
	char name[80];
	bool found=false;
	CAT *pHead, *pTemp;
	pHead=create(4);  // Create 4 CAT ojects

	pTemp=pHead;	// Set CATs names one by one
	pTemp->setName("Frisky");
	pTemp=pTemp->getNext();
	pTemp->setName("Felix");
	pTemp=pTemp->getNext();
	pTemp->setName("Dolly");
	pTemp=pTemp->getNext();
	pTemp->setName("James");

	cout <<"Please enter an arbitrary CAT name:\n";
	cin >> name;

	pTemp=pHead;
	do	// Check the existence of the CAT object with the given name
	{
		if (strcmp(name,pTemp->getName())==0)
		{
			found=true;
			break;
		}
		pTemp=pTemp->getNext();
	}
	while (pTemp !=0);

	if (found == true)
		cout<<name<<" is found!\n";
	else
		cout <<name<<" cannot be found!\n";

	return 0;
```

```
}

CAT * create(int n)
{ //Create a linked list of n CAT objects
      CAT *pH,*pT,*pL;
      int i;
      pH = new CAT;
      pL = pH;
      for (i=1; i<n; i++)
      {
            pT = new CAT;
            pL->setNext(pT);
            pL = pT;
      }
      return pH;
}



//Part 2
#include <iostream>
using namespace std;

class CAT
{
public:
      CAT() {pNext = 0;}            // Constructor - set $pNext to NULL by default
      void setName(char *nm)
      {                             // copy nm to $name and make sure there must
            strncpy (name,nm,79);   // be a NULL character at the end of name
            name[79] = NULL;
      }
      char * getName()              // get the name of a CAT
      {
            return name;
      }
      void setNext(CAT *pC)         // $pNext = pC
      {
            pNext = pC;
      }
      CAT * getNext()               // return $pNext
      {
            return pNext;
      }
private:
      char name[80];                // Keep the name of the CAT
      CAT * pNext;
};

CAT * insert (char * name, int i, CAT *pH);     //function prototypes
CAT * del (int i, CAT *pH);   //function prototypes

int main()
{
      char choice, name[80];
      int count, pos;

      bool exitflag=false;
      CAT *pHead=0, *pTemp;  //pHead initialized to NULL

      while (exitflag==false)
      {
            cout<<"\n\nA. Insert a CAT to the linked list\n";
            cout<<"B. Delete a CAT from the linked list\n";
            cout<<"C. Show all CAT objects\n";
            cout<<"Q. Quit.\n\n";
            cout<<"Your choice (A, B, C or Q):";
            cin >> choice;
```

```
            switch (choice)
            {
                  case 'A':
                        cout <<"It's name: ";
                        cin >> name;
                        cout <<"It's Position: ";
                        cin >> pos;
                        pHead= insert(name, pos, pHead);
                        break;

                  case 'B':
                        cout <<"It's Position: ";
                        cin >> pos;
                        pHead=del(pos, pHead);
                        break;

                  case 'C':
                        count=0;
                        pTemp=pHead;
                        while (pTemp !=0) //even no CAT in the list can be tackled
                        {
                              count++;
                              cout << "Cat "<< count<< ": "<< pTemp->getName() <<
endl;
                              pTemp=pTemp->getNext();
                        }
                        break;

                  case 'Q':
                        cout<< "Good bye!\n\n";
                        exitflag=true;
                        pTemp=pHead;
                        while (pTemp !=0)//delete all CAT objects
                        {
                              pHead=pTemp;
                              pTemp=pTemp->getNext();
                              delete pHead;
                        }
                        break;

                  default:
                        cout << "Wrong input\n\n"; break;
            } // end of switch - case
      } // end of while loop
return 0;
}



CAT * insert (char * name, int i, CAT *pHead)
{     //The first CAT object is at position 1
      CAT *pCurr, *pPrev, *pCat;
      bool inserted=false;
      int j=0;    //CatNum to be searched
      pCurr = pHead;
      if (i==1)         // Adding to the 1st position
      {
            pCat = new CAT;
            pCat->setNext(pCurr);// point the current CAT to be the next CAT
            pCat->setName(name); // write CatName
            pHead=pCat;              // point the new CAT to be the 1st CAT
            inserted = true;
      }
      else
      {
            do // find the item
            {
                  pPrev = pCurr;
```

```
                  j++;
                  pCurr = pCurr->getNext();
                  if (j==i-1)
                  {
                        pCat = new CAT;
                        pCat->setNext(pCurr);   // point the current CAT to be the
next CAT
                        pPrev->setNext(pCat);   // point the new CAT to be the
current CAT
                        pCat->setName(name); // write CatName
                        inserted= true;
                        break;
                  }
            }
            while (pCurr != 0);
      }
      if (inserted==false)
            cout<< "Cannot be inserted!\n";

      return pHead;
}


CAT * del(int n, CAT *pH) // to remove an item
{
      CAT *pCurr, *pPrev, *pCat;
      int i=0, count=1;
      if (pH==0)
      {
            cout << "Cannot be deleted!\n";
            return pH;
      }
      pCurr = pH;

      while (pCurr->getNext()!=0)   //count the number of CAT objects
      {
            count++;
            pCurr=pCurr->getNext();
      }

      if (n>count)
      {
            cout << "Cannot be deleted!\n";
            return pH;
      }

      pCurr = pH;
      if (n==1)         // To remove 1st object
            pH=pCurr->getNext();
      else
      {
            do // find the item
            {
                  pPrev = pCurr;
                  pCurr = pPrev->getNext();
                  i++;
            }
            while (i<n-1);

            pCat=pCurr->getNext();
            pPrev->setNext(pCat);
      }
      delete pCurr;
      return pH;
}
```