Linear Lists – Linked Allocation

Program 1:

```
//linkedlist.cpp
//first attempt at processing a linked list
//list is in no particular order; items are added at the front
#include <iostream>
//using namespace std;

class node{
public:

   void setData(int);
   int getData();
   void setNextPtr ();
   node* getNextPtr();
public: // This should be private; but this is a primitive first attept
    int data;
   node* nextPtr;
};

int main(){
    node* firstPtr = NULL;        // creates a new empty list
    int val;
   node* p;
   cout << "To end this program enter a negative number.\n\n";


   cout << "Enter a positive integer:  "; cin >> val;
   while (val >= 0) {            // read and add values to the list
    p = new node;
    p->data = val;
       p->nextPtr = firstPtr;
       firstPtr = p;
       cout << "Enter a positive integer:  "; cin >> val;
   }//end while
   cout << "\nNow I'll print out your list from first item to last.\n";
   p = firstPtr;
   while (p != 0){
    cout << p->data << endl;
       p = p->nextPtr;
   }
   char c; cin>>c; //holds console window
   return 0;
}
```

Program 2:

```
//linkedlist1.cpp
//first attempt at processing a linked list
//list is in no particular order; items are added at the front
#include <iostream>
//using namespace std;

class node{
```

```
public:

   void setData(int);
   int getData();
   void setNextPtr ();
   node* getNextPtr();
public: // This should be private; but this is a primitive first attempt
    int data;
   node* nextPtr;
};

int main(){
    node* firstPtr = NULL;        // creates a new empty list
    int val;
   node* p;

   //read and add values to the list
   cout << "To end this program enter a negative number.\n\n";
   cout << "Enter a positive integer:  "; cin >> val;
   while (val >= 0) {
    p = new node;
    p->data = val;
       p->nextPtr = firstPtr;
       firstPtr = p;
       cout << "Enter a positive integer:  "; cin >> val;
   }//end while

   //print all elements in the list first to last
   cout << "\nNow I'll print out your list from first item to last.\n";
   p = firstPtr;
   while (p != 0){
    cout << p->data << endl;
       p = p->nextPtr;
   }

   //sequential search of linked list
   char c;
   cout << "Do you wish to search for a target value? (y/n)" ; cin >> c;
   while (c != 'n'){
    cout << "\nEnter target:  "; cin >> val;
    p = firstPtr;
    while (p != NULL) {
                            if (p->data == val) break;
                            p = p->nextPtr;
    }
                            if (p == NULL) cout << "target not found" << endl;
                            else cout << "target found at location " << p <<
endl;

       cout << "Do you wish to search for a target value? (y/n)" ; cin >> c;
   }

   //end of program
   {char c; cin>>c;} //holds console window
   return 0;
}
```

Program:

```
//linkedlist2.cpp
```

```
//working with a linked list

#include <iostream>

class node{
public:
   //constructor, destructor
   void setData(int);   // get and set functions
   int getData();
   void setNextPtr ();
   node* getNextPtr();
public: // This should be private; but this is a primitive first attept
   int data;
   node* nextPtr;
};

int main(){
   node* firstPtr = NULL;        // creates a new empty list
   node* lastPtr = NULL;
   int val;
   node * p, * q, * pnew;

   // read and add new values to the list
   cout << "To end program enter a negative number. \n";
   cout << "Enter a positive integer:  "; cin >> val;
   while (val >= 0) {
   pnew = new node;
      pnew->data = val;

      if (firstPtr == NULL){     //list is empty
                           firstPtr = lastPtr = pnew;
         pnew->nextPtr = NULL;
         }
                           else {
                                  //search for position of new value in list
                           p = firstPtr;
                           while (p != NULL){
                           if (p->data > val) break;
                                  q = p;
                           p = p->nextPtr;
                                  }
                           //insert in proper position
                           if (p == firstPtr){        //new first
                                  pnew->nextPtr = firstPtr;
                           firstPtr = pnew;
                                  }
                           else if (p == NULL){       //new last
                                  pnew->nextPtr = NULL;
                           lastPtr->nextPtr = pnew;
                           lastPtr = pnew;
                           }
                           else {                              //insert before p
and after q
                                  pnew->nextPtr = p;
                           q->nextPtr = pnew;
                           }
            }
            cout << "Enter a positive integer:  "; cin >> val;
                           }

   //print all elements in the list first to last
   cout << "\nNow I'll print out your list from first item to last.\n";
   p = firstPtr;
```

```
   while (p != 0){
    cout << p->data << endl;
       p = p->nextPtr;
    } //

   //end of program
   char c; cin>>c; //holds console window
   return 0;
}   //
```

Program:

```
//linkedlist3.cpp
//first attempt at processing a linked list
//list is in no particular order; items are added at the front
#include <iostream>
//using namespace std;

class node{
public:
    //constructor, destructor funs
   void setData(int);
   int getData();
   void setNextPtr ();
   node* getNextPtr();
public: // This should be private; but this is a primitive first attempt
    int data;
   node* nextPtr;
};

void printup (node*);
void printdown (node*);

int main(){
    node* firstPtr = NULL;        // creates a new empty list
    int val;
   node* p;

   //read and add values to the list
   cout << "To end this program enter a negative number.\n\n";
   cout << "Enter a positive integer:  "; cin >> val;
   while (val >= 0) {
    p = new node;
    p->data = val;
       p->nextPtr = firstPtr;
       firstPtr = p;
       cout << "Enter a positive integer:  "; cin >> val;
   }//end while

   //print all elements in the list first to last
   cout << "\nNow I'll print out your list from first item to last.\n";
   printup(firstPtr);

   //print all elements in the list last to first
   cout << "\nNow I'll print out your list from last item to first.\n";
   printdown(firstPtr);


   //end of program
   {char c; cin>>c;} //holds console window
```

```
   return 0;
}


void printup (node* p){
   if (p == NULL) return;
   else {
                           cout << p->data << endl;
                           printup (p->nextPtr);
      }
}

void printdown (node* p){
   if (p == NULL) return;
   else {
                           printdown (p->nextPtr);
         cout << p->data << endl;
      }
}
```