

```
// Vishnu Prasad Vishwanathan  
// 793782749  
// vvishwan@syr.edu
```

```
#include "stdafx.h"  
#include <iostream>  
using namespace std;  
  
class node {  
public:  
    int value;  
    node * next;  
    node() { next = nullptr; }  
    node(int k) { value = k; next = nullptr; }  
};  
  
class nexted_list {  
public:  
    int num_nodes;  
    node * head;  
    nexted_list() { num_nodes = 0; head = nullptr; }  
    void make_random_list(int k);  
    void print();  
    void reverse();  
    void bubble_sort();  
    void selection_sort();  
  
    void insertion_sort();  
};  
  
void nexted_list::make_random_list(int k) {  
    node * p;
```

```

        for (int i = 0; i < k; i++) {
            p = new node(rand() % 100);
            p->next = head;
            head = p;
            num_nodes++;
        }
    }

void nexted_list::print() {
    node * p = head;
    cout << endl;
    while (p != nullptr) {
        cout << p->value << " ";
        p = p->next;
    }
}

void nexted_list::reverse() {
    if (num_nodes <= 1) return;
    node * p1 = head, *p2 = head->next, *p3;
    while (p2 != nullptr) {
        p3 = p2->next;
        p2->next = p1;
        if (p1 == head) p1->next = nullptr;
        p1 = p2;
        p2 = p3;
    }
    head = p1;
}

void nexted_list::bubble_sort() {
    int i = 0;
    node * p1 = head, *p2 = nullptr, *p3 = nullptr;
    bool swap_done = true;

    while (swap_done) { p1 =
        head; swap_done =
        false;
        while ((p1->next) != p3)
        {
            if (p1->value > (p1->next)->value)
            {
                i = p1->value;
                p1->value = p1->next->value;
                p1->next->value = i;

                swap_done = true;
            }
            p1 = p1->next;
        } p3 = p1;
    }
}

void nexted_list::insertion_sort()
{
    int i = 0;
    node * p1 = new node();
    p1->next = head;
    head = p1;
}

```

```

node * p2 = new node();
node * p3 = new node();

bool swap_done1 = false;
while (p1->next != NULL)
{
    p2 = head;
    swap_done1 = false;
    while (p2 != p1)
    {
        if (p2->next->value > p1->next->value)
        {
            p3 = p1->next;
            p1->next = p1->next->next;
            p3->next = p2->next;
            p2->next = p3;
            swap_done1 = true;
            break;
        }
        else
            p2 = p2->next;
    }

    if (!swap_done1) {
        p1 = p1->next;
    }
}

void nexted_list::selection_sort()
{
    int number;
    node *p1, *p2;
    for (p1 = head; p1 != NULL; p1 = p1->next) {
        for (p2 = p1->next; p2 != NULL; p2 = p2->next) {
            if (p1->value > p2->value) {
                number = p1->value;
                p1->value = p2->value;
                p2->value = number;
            }
        }
    }
}

int main() {

    cout << "Input List";
    cout << "-----";
    nexted_list L1;
    L1.make_random_list(50);
    L1.print();
    cout << "\n-----\n";
    cout << "\nBubble Sort Output\n";
    cout << "\n-----\n";
    L1.bubble_sort();
    L1.print();

    cout << "\nInput List";

```

```

cout << "\n-----\n";
nexted_list L2;
L2.make_random_list(50);
L2.print();
cout << "\n-----\n";
cout << "\nSelection Sort Output\n";
cout << "\n-----\n";
L2.selection_sort();
L2.print();

cout << "\nInput List\n";
cout << "\n-----\n";
nexted_list L3;
L3.make_random_list(50);
L3.print();
cout << "\n-----\n";
cout << "\nInsertion Sort Output\n";
cout << "\n-----\n";
L3.insertion_sort();
L3.print();

getchar();
getchar();
return 0;
}

```