Đã bắt đầu vào lúc	Thứ sáu, 20 Tháng mười 2023, 9:12 PM
Tình trạng	Đã hoàn thành
Hoàn thành vào lúc	Thứ bảy, 21 Tháng mười 2023, 3:47 PM
Thời gian thực hiện	18 giờ 35 phút
Điểm	3,00/3,00
Điểm	**10,00** của 10,00 (**100**%)

Câu hỏi 1

Chính xác

Điểm 1,00 của 1,00

Implement method bubbleSort() in class SLinkedList to sort this list in ascending order. After each bubble, we will print out a list to check (using printList).

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

template <class T>
class SLinkedList {
public:
    class Node; // Forward declaration
protected:
    Node* head;
    Node* tail;
    int count;
public:
    SLinkedList()
    {
      this->head = nullptr;
      this->tail = nullptr;
      this->count = 0;
    }
    ~SLinkedList(){};
    void add(T e)
    {
        Node *pNew = new Node(e);

        if (this->count == 0)
        {
            this->head = this->tail = pNew;
        }
        else
        {
            this->tail->next = pNew;
            this->tail = pNew;
        }

        this->count++;
    }
    int size()
    {
        return this->count;
    }
    void printList()
    {
        stringstream ss;
        ss << "[";
        Node *ptr = head;
        while (ptr != tail)
        {
            ss << ptr->data << ",";
            ptr = ptr->next;
        }

        if (count > 0)
            ss << ptr->data << "]";
        else
            ss << "]";
        cout << ss.str() << endl;
    }
public:
    class Node {
    private:
        T data;
        Node* next;
        friend class SLinkedList<T>;
    public:
        Node() {
```

```
            next = 0;
        }
        Node(T data) {
            this->data = data;
            this->next = nullptr;
        }
    };

    void bubbleSort();
};
```

For example:

Test	Result

int arr[] = {9, 2, 8, 4, 1};
SLinkedList<int> list;
for(int i = 0; i <int(sizeof(arr))/4;i++)
list.add(arr[i]);
list.bubbleSort();
	[2,8,4,1,9]
	[2,4,1,8,9]
	[2,1,4,8,9]
	[1,2,4,8,9]

Answer: (penalty regime: 0 %)

Reset answer

```
template <class T>
void SLinkedList<T>::bubbleSort()
{
    int current = 0;
    bool flag = false;
    while(current < this->count && flag == false) {
        flag = true;
        Node* cur = head;
        while(cur != tail) {
            if(cur->data > cur->next->data) {
                flag = false;
                T val = cur->data;
                cur->data = cur->next->data;
                cur->next->data = val;
            }
            cur = cur->next;
        }
        current++;
        if(!flag) this->printList();
    }
}
```

	Test	Expected	Got	

✔	`int arr[] = {9, 2, 8, 4, 1};`
	`SLinkedList<int> list;`
	`for(int i = 0; i <int(sizeof(arr))/4;i++)`
	`    list.add(arr[i]);`
	`list.bubbleSort();`
		[2,8,4,1,9]
		[2,4,1,8,9]
		[2,1,4,8,9]
		[1,2,4,8,9]
			[2,8,4,1,9]
			[2,4,1,8,9]
			[2,1,4,8,9]
			[1,2,4,8,9]
				✔

Passed all tests! ✔

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 2

Chính xác

Điểm 1,00 của 1,00

Implement static method selectionSort in class **Sorting** to sort an array in ascending order. After each selection, we will print out a list to check (using printArray).

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

template <class T>
class Sorting
{
public:
    /* Function to print an array */
    static void printArray(T *start, T *end)
    {
        int size = end - start;
        for (int i = 0; i < size - 1; i++)
            cout << start[i] << ", ";
        cout << start[size - 1];
        cout << endl;
    }

    static void selectionSort(T *start, T *end);
};
```

For example:

Test	Result
int arr[] = {9, 2, 8, 1, 0, -2}; Sorting<int>::selectionSort(&arr[0], &arr[6]);	-2, 2, 8, 1, 0, 9 -2, 0, 8, 1, 2, 9 -2, 0, 1, 8, 2, 9 -2, 0, 1, 2, 8, 9 -2, 0, 1, 2, 8, 9

Answer: (penalty regime: 0 %)

Reset answer

```
template <class T>
void Sorting<T>::selectionSort(T *start, T *end)
{
    int length = end - start;
    for(int i = 0; i < length - 1; i++) {
        int smallest = i;
        for(int j = i+1; j < length; j++) {
            if(start[j] < start[smallest]) smallest = j;
        }
        swap(start[smallest], start[i]);
        Sorting<T>::printArray(start,end);
    }
}
```

	Test	Expected	Got	

✔	`int arr[] = {9, 2, 8, 1, 0, -2};`
	`Sorting<int>::selectionSort(&arr[0], &arr[6]);`
		`-2, 2, 8, 1, 0, 9`
		`-2, 0, 8, 1, 2, 9`
		`-2, 0, 1, 8, 2, 9`
		`-2, 0, 1, 2, 8, 9`
		`-2, 0, 1, 2, 8, 9`
			`-2, 2, 8, 1, 0, 9`
			`-2, 0, 8, 1, 2, 9`
			`-2, 0, 1, 8, 2, 9`
			`-2, 0, 1, 2, 8, 9`
			`-2, 0, 1, 2, 8, 9`
				✔

Passed all tests! ✔

Chính xác

Điểm cho bài nộp này: 1,00/1,00.

Câu hỏi 3

Chính xác

Điểm 1,00 của 1,00

Implement static methods **sortSegment** and **ShellSort** in class **Sorting** to sort an array in ascending order.

```
#ifndef SORTING_H
#define SORTING_H

#include <sstream>
#include <iostream>
#include <type_traits>
using namespace std;

template <class T>
class Sorting {
private:
    static void printArray(T* start, T* end)
    {
        int size = end - start;
        for (int i = 0; i < size; i++)
            cout << start[i] << " ";
        cout << endl;
    }

public:
    // TODO: Write your code here
    static void sortSegment(T* start, T* end, int segment_idx, int cur_segment_total);
    static void ShellSort(T* start, T* end, int* num_segment_list, int num_phases);
};

#endif /* SORTING_H */
```

For example:

Test	Result

`int num_segment_list[] = {1, 3, 5};`
`int num_phases = 3;`
`int array[] = { 10, 9, 8 , 7 , 6, 5, 4, 3, 2, 1 };`
 	
`Sorting<int>::ShellSort(&array[0], &array[10], &num_segment_list[0], num_phases);`
	`5 segments: 5 4 3 2 1 10 9 8 7 6`
	`3 segments: 2 1 3 5 4 7 6 8 10 9`
	`1 segments: 1 2 3 4 5 6 7 8 9 10`

Answer: (penalty regime: 0 %)

Reset answer

```
1  // TODO: Write your code here
2
3  static void sortSegment(T* start, T* end, int segment_idx, int cur_segment_total) {
4      // TODO
5      int current  = segment_idx + cur_segment_total;
6      while(current < end - start) {
7          T key = start[current];
8          int step = current - cur_segment_total;
9          while(step >= 0 && key < start[step]) {
10             start[step + cur_segment_total] = start[step];
11             step = step - cur_segment_total;
12         }
13         start[step + cur_segment_total] = key;
14         current += cur_segment_total;
15     }
16 }
17
18 static void ShellSort(T* start, T* end, int* num_segment_list, int num_phases) {
19     // TODO
```

```
    // Note: You must print out the array after sorting segments to check whether your algorithm is true.
    if(start - end == 1) {
        Sorting<T>::printArray(start, end);
        return;
    }
    for(int i = num_phases-1; i >= 0; i--) {
        int segment = 0;
        while(segment < num_segment_list[i]) {
            sortSegment(start,end, segment, num_segment_list[i]);
            segment += 1;
        }
        cout << num_segment_list[i] << " segments: ";
        Sorting<T>::printArray(start, end);
    }
}
```

	Test	Expected	Got	

✔	`int num_segment_list[] = {1, 3, 5};`
	`int num_phases = 3;`
	`int array[] = { 10, 9, 8 , 7 , 6, 5, 4, 3, 2, 1 };`
	
	`Sorting<int>::ShellSort(&array[0], &array[10], &num_segment_list[0], num_phases);`
		5 segments: 5 4 3 2 1 10 9 8 7 6
		3 segments: 2 1 3 5 4 7 6 8 10 9
		1 segments: 1 2 3 4 5 6 7 8 9 10
			5 segments: 5 4 3 2 1 10 9 8 7 6
			3 segments: 2 1 3 5 4 7 6 8 10 9
			1 segments: 1 2 3 4 5 6 7 8 9 10
				✔

✔	`int num_segment_list[] = { 1, 2, 6 };`
	`int num_phases = 3;`
	`int array[] = { 10, 9, 8 , 7 , 6, 5, 4, 3, 2, 1 };`
	
	`Sorting<int>::ShellSort(&array[0], &array[10], &num_segment_list[0], num_phases);`
		6 segments: 4 3 2 1 6 5 10 9 8 7
		2 segments: 2 1 4 3 6 5 8 7 10 9
		1 segments: 1 2 3 4 5 6 7 8 9 10
			6 segments: 4 3 2 1 6 5 10 9 8 7
			2 segments: 2 1 4 3 6 5 8 7 10 9
			1 segments: 1 2 3 4 5 6 7 8 9 10
				✔

✔	`int num_segment_list[] = { 1, 2, 5 };`
	`int num_phases = 3;`
	`int array[] = { 10, 9, 8 , 7 , 6, 5, 4, 3, 2, 1 };`
	
	`Sorting<int>::ShellSort(&array[0], &array[10], &num_segment_list[0], num_phases);`
		5 segments: 5 4 3 2 1 10 9 8 7 6
		2 segments: 1 2 3 4 5 6 7 8 9 10
		1 segments: 1 2 3 4 5 6 7 8 9 10
			5 segments: 5 4 3 2 1 10 9 8 7 6
			2 segments: 1 2 3 4 5 6 7 8 9 10
			1 segments: 1 2 3 4 5 6 7 8 9 10
				✔

✔	`int num_segment_list[] = { 1, 2, 3 };`
	`int num_phases = 3;`
	`int array[] = { 10, 9, 8 , 7 , 6, 5, 4, 3, 2, 1 };`
	
	`Sorting<int>::ShellSort(&array[0], &array[10], &num_segment_list[0], num_phases);`
		3 segments: 1 3 2 4 6 5 7 9 8 10
		2 segments: 1 3 2 4 6 5 7 9 8 10
		1 segments: 1 2 3 4 5 6 7 8 9 10
			3 segments: 1 3 2 4 6 5 7 9 8 10
			2 segments: 1 3 2 4 6 5 7 9 8 10
			1 segments: 1 2 3 4 5 6 7 8 9 10
				✔

✔	`int num_segment_list[] = { 1, 5, 8, 10 };`
	`int num_phases = 4;`
	`int array[] = { 3, 5, 7, 10 ,12, 14, 15, 13, 1, 2, 9, 6, 4, 8, 11 };`
	
	`Sorting<int>::ShellSort(&array[0], &array[15], &num_segment_list[0], num_phases);`
		10 segments: 3 5 4 8 11 14 15 13 1 2 9 6 7 10 12
		8 segments: 1 2 4 6 7 10 12 13 3 5 9 8 11 14 15
		5 segments: 1 2 4 3 5 9 8 11 6 7 10 12 13 14 15
		1 segments: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
			10 segments: 3 5 4 8 11 14 15 13 1 2 9 6 7 10 12
			8 segments: 1 2 4 6 7 10 12 13 3 5 9 8 11 14 15
			5 segments: 1 2 4 3 5 9 8 11 6 7 10 12 13 14 15
			1 segments: 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15
				✔

Passed all tests! ✔

Chính xác

Điểm cho bài nộp này: 1,00/1,00.