Answers Question

+ ផ្នែកសំណួរ

១.ចូរគូរផ្គងពាក្យខាងឆ្វេង និងនិយមន័យដែលត្រឹមត្រូវខាងស្តាំ

១	២	៣	៤	៥	៦	៧	៨	៩	១០
ច	ឃ	ក	ឆ	ជ	ញ	ខ	ង	ឈ	គ

២.ចូរសរសេរពាក្យ True ប្រសិនបើប្រយោគនេះត្រឹមត្រូវនិងសរសេរពាក្យថា False ប្រើសិនបើប្រយោគនេះមិនត្រឹមត្រូវ

1	2	3	4	5	6	7	8	9	10
ក	គ	ង	គ	ឃ	ខ	ក	ឃ	ក	ង

៣.ចូរជ្រើសរើសចម្លើយដែលត្រឹមត្រូវបំផុតសម្រាបប្រយោគនីមួយៗ

1	2	3	4	5	6	7	8	9	10
ក	ឃ	ក	ឃ	ង	ក	ង	ក	ខ	គ

៤.ផ្នែកលំហាត់

១.ចូរប្អូនសរសេរ function សម្រាប់ធ្វើការលុបធាតុចេញពី Link List

- Function សម្រាប់លុបធាតុដំបូងករណី Link List មានតែមួយធាតុ

```
// Function to delete first index
    void deleteFirst() {
        if (head) {
            Node* temp = head;
            head = head->next;
            delete temp;
        } else {
            std::cout << "List is empty. Cannot delete from an empty list." << std::endl;
        }
    }
```

- Function សម្រាប់លុបធាតុដំបូងករណី Link List មានច្រើនធាតុ

```
// Function to delete many index
void deleteAtIndex(int index) {
    if (index < 0) {
        std::cout << "Invalid index. Index must be non-negative." << std::endl;
        return;
    }

    if (index == 0) {
        if (head) {
            Node* temp = head;
            head = head->next;
            delete temp;
        } else {
            std::cout << "List is empty. Cannot delete from an empty list." << std::endl;
        }
    } else {
        Node* current = head;
        for (int i = 0; i < index - 1 && current; ++i) {
            current = current->next;
        }
        if (current && current->next) {
            Node* temp = current->next;
            current->next = temp->next;
            delete temp;
        } else {
            std::cout << "Invalid index. Index out of range." << std::endl;
        }
    }
}
```

២.ចូរប្អូនសរសេរ function សម្រាប់ធ្វើការលុបធាតុចេញពី Link List

- Function សម្រាប់លុបធាតុចុងក្រោយករណី Link List មានច្រើនធាតុ

```
// Function to delete the last node
    void deleteLast() {
        if (!head) {
            std::cout << "List is empty. Cannot delete from an empty list." << std::endl;
            return;
        }
        // If there is only one node in the list
        if (!head->next) {
            delete head;
            head = nullptr;
            return;
        }
        Node* current = head;
        Node* prev = nullptr;
        while (current->next) {
            prev = current;
            current = current->next;
        }
        if (prev) {
            prev->next = nullptr;
        } else {
            // If there are only two nodes in the list
            head = nullptr;
        }
        delete current;
    }
```

- Function សម្រាប់លុបធាតុត្រង់ចន្លោះណាមួយករណី Link List មានច្រើនធាតុ

```
// Function to delete the center node (assuming odd number of nodes)
    void deleteAtCenter() {
        if (!head) {
            std::cout << "List is empty. Cannot delete from an empty list." << std::endl;
            return;
        }

        Node* slow = head;
        Node* fast = head;
        Node* prev = nullptr;

        while (fast && fast->next) {
            prev = slow;
            slow = slow->next;
            fast = fast->next->next;
        }

        if (prev) {
            prev->next = slow->next;
        } else {
            // If there is only one node in the list
            head = slow->next;
        }

        delete slow;
    }
```