
1- Write a program to store contacts (names and phone numbers) using a hash table.

اكتب برنامجًا لتخزين جهات الاتصال (الأسماء وأرقام الهواتف) باستخدام hash table.

Input & Output

```
Name: name1
Number: 012345
Do you have another contact? yes
Name: name2
Number: 15987
Do you have another contact? yes
Name: name3
Number: 23654
Do you have another contact? no
Contacts:
Name: name1
Number: 012345
Name: name2
Number: 15987
Name: name3
Number: 23654
```

Solution

```cpp
// www.gammal.tech

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

struct Contact {
    string name;
    string number;
    Contact* next;
};

int hashCode(char c) {
    if (c >= 'a' && c <= 'z')
        return (int)c - 'a' + 1;
    return (int)c - 'A' + 1;
}

int hashFunction(string x) {
    int hash = 0;
    for (int i = 1; i <= x.size(); i++)
        hash += hashCode(x[i - 1]) * i;
    return hash;
}

Contact* hashTable[100] = {nullptr};

void insert(int index, Contact* c) {
    if (hashTable[index % 100] == nullptr) {
        hashTable[index % 100] = c;
        return;
    }
    Contact* p = hashTable[index % 100];
    while (p->next != nullptr)
        p = p->next;
    p->next = c;
}

int main() {
    string ans;
    do {
        Contact* c = new Contact;
        cout << "Name: ";
        cin >> c->name;
        cout << "Number: ";
        cin >> c->number;
        c->next = nullptr;
        int index = hashFunction(c->name);
        insert(index, c);
        cout << "Do you have another contact? ";
        cin >> ans;
    } while (ans == "yes");

    cout << "Contacts:" << endl;
    for (int i = 0; i < 100; i++) {
        Contact* p = hashTable[i];
        while (p != nullptr) {
            cout << "Name: " << p->name << endl;
            cout << "Number: " << p->number << endl;
            p = p->next;
        }
    }
    return 0;
}
```

2- Write a program to search for a contact by name in the hash table.

اكتب برنامجًا للبحث عن جهة اتصال بالاسم في hash table.

Input & Output

```
Enter name: name1
Enter number: 1234
Do you have another contact? (y/n): y
Enter name: name2
Enter number: 357
Do you have another contact? (y/n): y
Enter name: name3
Enter number: 36851
Do you have another contact? (y/n): n
Enter the name to search: name2
Contact found - Name: name2, Number: 357
```

Solution

```cpp
// www.gammal.tech

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

struct Contact {
    string name;
    string number;
    Contact* next;
};

const int TABLE_SIZE = 100;
Contact* hashTable[TABLE_SIZE] = {nullptr};

int hashCode(const string& str) {
    int hash = 0;
    for (char c : str) {
        hash += c;
    }
    return hash % TABLE_SIZE;
}

void insert(const string& name, const string& number) {
    int index = hashCode(name);
    Contact* newContact = new Contact;
    newContact->name = name;
    newContact->number = number;
    newContact->next = nullptr;

    if (hashTable[index] == nullptr) {
        hashTable[index] = newContact;
    } else {
        Contact* current = hashTable[index];
        while (current->next != nullptr) {
            current = current->next;
        }
        current->next = newContact;
    }
}

Contact* search(const string& name) {
    int index = hashCode(name);
    Contact* current = hashTable[index];
    while (current != nullptr) {
        if (current->name == name) {
            return current;
        }
        current = current->next;
    }
    return nullptr;
}

int main() {
    string name, number;
    char ans;
    do {
        cout << "Enter name: ";
        cin >> name;
        cout << "Enter number: ";
        cin >> number;
        insert(name, number);
        cout << "Do you have another contact? (y/n): ";
        cin >> ans;
    } while (ans == 'y');

    cout << "Enter the name to search: ";
    cin >> name;
    Contact* result = search(name);
    if (result != nullptr) {
        cout << "Contact found - Name: " << result->name << ", Number: " << result->number << endl;
    } else {
        cout << "Contact not found." << endl;
    }

    return 0;
}
```

3- Create a program to count the frequency of each letter in a given string provided by the user.

إنشاء برنامج لحساب تكرار كل حرف في سلسلة معينة يقدمها المستخدم.

Input

```
Enter a string: Hello
```

Output

```
Letter frequencies:
l: 2
e: 1
o: 1
H: 1
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <unordered_map>
#include <string>
using namespace std;

int main() {
    string str;
    cout << "Enter a string: ";
    getline(cin, str);

    unordered_map<char, int> freq;
    for (char ch : str) {
        freq[ch]++;
    }

    cout << "Letter frequencies:" << endl;
    for (auto pair : freq) {
        cout << pair.first << ": " << pair.second << endl;
    }

    return 0;
}
```

4- Create a program to reverse a given string provided by the user.

قم بإنشاء برنامج لعكس سلسلة معينة يقدمها المستخدم.

Input

```
Enter a string: Hello
```

Output

```
Reversed string: olleH
```

Solution

```cpp
// www.gammal.tech

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

string reverseString(const string& str) {
    string reversed;
    for (int i = str.size() - 1; i >= 0; i--) {
        reversed += str[i];
    }
    return reversed;
}

int main() {
    string str;
    cout << "Enter a string: ";
    cin >> str;

    cout << "Reversed string: " << reverseString(str) << endl;

    return 0;
}
```

5- Create a program to check if a given string provided by the user is a palindrome.

أنشئ برنامجًا للتحقق مما إذا كانت السلسلة المعطاة التي يقدمها المستخدم متناظرة.

Input

```
Enter a string: llell
```

Output

```
The string is a palindrome.
```

Solution

```cpp
// www.gammal.tech

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

bool isPalindrome(const string& str) {
    int left = 0, right = str.size() - 1;
    while (left < right) {
        if (str[left] != str[right])
            return false;
        left++;
        right--;
    }
    return true;
}

int main() {
    string str;
    cout << "Enter a string: ";
    cin >> str;

    if (isPalindrome(str))
        cout << "The string is a palindrome." << endl;
    else
        cout << "The string is not a palindrome." << endl;

    return 0;
}
```

6- Create a program to find the first non-repeating character in a given string provided by the user.

قم بإنشاء برنامج للعثور على الحرف الأول غير المتكرر في سلسلة معينة يوفرها المستخدم.

Input

```
Enter a string: Hello
```

Output

```
First non-repeating character: H
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
#include <string>
#include <unordered_map>
using namespace std;

char firstNonRepeating(const string& str) {
    unordered_map<char, int> freq;
    for (char ch : str) {
        freq[ch]++;
    }
    for (char ch : str) {
        if (freq[ch] == 1)
            return ch;
    }
    return '\0';
}

int main() {
    string str;
    cout << "Enter a string: ";
    cin >> str;

    char result = firstNonRepeating(str);
    if (result != '\0')
        cout << "First non-repeating character: " << result << endl;
    else
        cout << "No non-repeating character found." << endl;

    return 0;
}
```

7- Write a program to find and print the average of all numbers in the linked list using recursion.

اكتب برنامجًا للعثور على متوسط جميع الأرقام في linked list وطباعته using recursion.

Input

```
Enter a number that represents the number of numbers in the linked list: 4
Enter number: 1
Enter number: 2
Enter number: 3
Enter number: 4
```

Output

```
Average of numbers: 2.50
```

Solution

```c
// www.gammal.tech

#include <stdio.h>
#include <stdlib.h>

struct gammal {
    int number;
    struct gammal* next;
};

struct gammal* add(struct gammal* g) {
    if (g == NULL) {
        g = (struct gammal*)malloc(sizeof(struct gammal));
        printf("Enter number: ");
        scanf("%d", &g->number);
        g->next = NULL;
        return g;
    }
    g->next = add(g->next);
    return g;
}

float findAverage(struct gammal* g, int sum, int count) {
    if (g == NULL) {
        return (float)sum / count;
    }
    return findAverage(g->next, sum + g->number, count + 1);
}

int main() {
    struct gammal* head = NULL;
    int numberSelect;

    printf("Enter a number that represents the number of numbers in the linked list: ");
    scanf("%d", &numberSelect);

    for (int i = 0; i < numberSelect; i++)
        head = add(head);

    printf("Average of numbers: %.2f\n", findAverage(head, 0, 0));

    return 0;
}
```

8- Write a program to find and print the product of all numbers in the linked list using recursion.

اكتب برنامجًا للعثور على ضرب جميع الأرقام في linked list وطباعته using recursion

Input

```
Enter a number that represents the number of numbers in the
linked list: 4
Enter number: 1
Enter number: 2
Enter number: 3
Enter number: 4
```

Output

```
Product of numbers: 24
```

Solution

```c
// www.gammal.tech

#include <stdio.h>
#include <stdlib.h>

struct gammal {
    int number;
    struct gammal* next;
};

struct gammal* add(struct gammal* g) {
    if (g == NULL) {
        g = (struct gammal*)malloc(sizeof(struct gammal));
        printf("Enter number: ");
        scanf("%d", &g->number);
        g->next = NULL;
        return g;
    }
    g->next = add(g->next);
    return g;
}

int findProduct(struct gammal* g) {
    if (g == NULL) {
        return 1; // Initialize with 1 for multiplication
    }
    return g->number * findProduct(g->next);
}

int main() {
    struct gammal* head = NULL;
    int numberSelect;

    printf("Enter a number that represents the number of numbers in the linked list: ");
    scanf("%d", &numberSelect);

    for (int i = 0; i < numberSelect; i++)
        head = add(head);

    printf("Product of numbers: %d\n", findProduct(head));

    return 0;
}
```

9- Write a C++ program to construct a binary search tree (BST) based on user input scores and then print the sum of all even numbers in the BST.

اكتب برنامج C++ لإنشاء شجرة بحث ثنائية (BST) استنادًا إلى درجات إدخال المستخدم، ثم اطبع مجموع جميع الأرقام الزوجية في شجرة البحث الثنائية.

Input & Output

```
1) Add score
2) Print sum of even numbers
3) Exit
Enter your choice: 1
Enter score: 1
1) Add score
2) Print sum of even numbers
3) Exit
Enter your choice: 1
Enter score: 2
1) Add score
2) Print sum of even numbers
3) Exit
Enter your choice: 1
Enter score: 3
1) Add score
2) Print sum of even numbers
3) Exit
Enter your choice: 1
Enter score: 4
1) Add score
2) Print sum of even numbers
3) Exit
Enter your choice: 1
Enter score: 5
1) Add score
2) Print sum of even numbers
3) Exit
Enter your choice: 2
Sum of even numbers in the BST: 6
1) Add score
2) Print sum of even numbers
3) Exit
Enter your choice: 3
Exiting the program...
```

Solution

```cpp
// www.gammal.tech

#include <iostream>
using namespace std;

// Structure definition for binary search tree (BST) node
struct Node {
    int score;
    Node* left;
    Node* right;

    // Constructor
    Node(int s) {
        score = s;
        left = nullptr;
        right = nullptr;
    }
};

// Function to add a score to the binary search tree (BST)
Node* add(Node* root, int score) {
    if (root == nullptr) {
        root = new Node(score);
        return root;
    }
    if (score < root->score)
        root->left = add(root->left, score);
    else if (score > root->score)
        root->right = add(root->right, score);

    return root;
}

// Function to calculate the sum of all even numbers in the BST
int sumEvenNumbers(Node* root) {
    if (root == nullptr)
        return 0;
    int sum = 0;
    if (root->score % 2 == 0)
        sum += root->score;
    sum += sumEvenNumbers(root->left);
    sum += sumEvenNumbers(root->right);
    return sum;
}

int main() {
    Node* root = nullptr;
    int choice;
    do {
        cout << "1) Add score\n2) Print sum of even numbers\n3) Exit\n";
        cout << "Enter your choice: ";
        cin >> choice;
        switch (choice) {
            case 1:
                int score;
                cout << "Enter score: ";
                cin >> score;
                root = add(root, score);
                break;
            case 2:
                cout << "Sum of even numbers in the BST: " << sumEvenNumbers(root) << endl;
                break;
            case 3:
                cout << "Exiting the program..." << endl;
                break;
            default:
                cout << "Invalid choice! Please enter again." << endl;
        }
    } while (choice != 3);
    return 0;
}
```

10- Write a C++ program that takes a sentence as input, stores each word in a linked list, and then outputs the words stored in the linked list.

اكتب برنامج C++ الذي يأخذ جملة كمدخل، ويخزن كل كلمة في linked list، ثم يخرج الكلمات المخزنة في linked list.

Output

```
Words stored in the linked list: I have mastered the data structure
```

Solution

```cpp
// www.gammal.tech

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

// Define structure for linked list node
struct Node {
    string data;
    Node* next;
};

int main() {
    // Input sentence
    string sentence = "I have mastered the data structure";

    // Create a stringstream object to extract words
    stringstream ss(sentence);
    string word;

    // Initialize head pointer for linked list
    Node* head = nullptr;
    Node* tail = nullptr;

    // Extract words and store them in linked list
    while (ss >> word) {
        // Create a new node
        Node* newNode = new Node;
        newNode->data = word;
        newNode->next = nullptr;

        // If list is empty, set head to new node
        if (head == nullptr) {
            head = newNode;
            tail = newNode;
        }
        // Otherwise, append new node to end of list
        else {
            tail->next = newNode;
            tail = newNode;
        }
    }

    // Output message
    cout << "Words stored in the linked list: ";

    // Traverse the linked list and print words
    Node* current = head;
    while (current != nullptr) {
        cout << current->data << " ";
        current = current->next;
    }
    cout << endl;

    // Free memory by deleting nodes
    current = head;
    while (current != nullptr) {
        Node* temp = current;
        current = current->next;
        delete temp;
    }

    return 0;
}
```