



REPUBLIC OF PHILIPPINES

BICOL UNIVERSITY

POLANGUI

Polangui, Albay



Jaiden Nykluz M. Fermante

Nicole Anne F. Bellen

Akisha miel Reomalis

BSIS-2B

DATA STRUCTURE AND ALGORITHM

PROJECT AND FINAL EXAMINATIONS

i. TITLE: Trivia Game

ii. THEME: The theme of the code is a trivia game implemented using different tree and heap data structures.

iii. PROJECT OVERVIEW: The game asks the user various trivia questions, and the questions are stored and accessed differently based on the underlying data structure being used:

Binary Tree: A simple tree where questions and answers are organized, and the user is guided through the tree based on whether they answer correctly or incorrectly.

Binary Search Tree (BST): Questions are inserted in lexicographical order, with the tree maintaining a structure where left subtrees contain lexicographically smaller questions and right subtrees contain larger ones.

Heap: A max-heap is used where questions are prioritized based on their difficulty level, ensuring that harder questions come first.

The theme incorporates:

Knowledge testing through trivia questions.

Data structures (Binary Tree, BST, Heap) to organize and traverse trivia questions in different ways.

User interaction where the user answers questions, and the game responds based on correctness.

Each section of the game provides a slightly different experience depending on the tree or heap structure used, demonstrating different ways of managing and accessing data in a game context.

iv. CODE

```
#include <iostream>
#include <string>
#include <vector>
#include <algorithm>
using namespace std;
//Binary Tree Implementation
class BinaryTree {
public:
    struct Node {
        string question; // Holds the trivia question.
        string answer; // Holds the correct answer for the question.
        Node* left; // Pointer to the left child.
        Node* right; // Pointer to the right child.
        Node(string q, string a) : question(q), answer(a), left(nullptr), right(nullptr) {}
    };
    Node* root;
    BinaryTree() : root(nullptr) {}
    // Inserts a question and answer into the binary tree.
    void insert(Node*& node, string question, string answer) {
        if (!node) {
            node = new Node(question, answer); // Create a new node if current node is null.
            return;
        }
        if (!node->left) // If left child is empty, insert there.
            insert(node->left, question, answer);
        else // Otherwise, insert into the right child.
            insert(node->right, question, answer);
    }
}
```

```

// Recursive function to play the trivia game using binary tree traversal.
void playGame(Node* node) {
    if (!node) return; // Base case: If node is null, exit.
    // Ask the current question.
    cout << node->question << "\nYour Answer: ";
    string userAnswer;
    getline(cin, userAnswer);
    transform(userAnswer.begin(), userAnswer.end(), userAnswer.begin(), ::tolower);
    string correctAnswer = node->answer;
    transform(correctAnswer.begin(), correctAnswer.end(), correctAnswer.begin(),
    ::tolower);
    if (userAnswer == correctAnswer) {
        cout << "Correct!\n\n";
        if (node->left) // Continue with the left child if present.
            playGame(node->left);
        else
            cout << "End of Binary Tree Trivia.\n\n"; // End if no left child.
    } else {
        cout << "Wrong! The correct answer was: " << node->answer << "\n\n";
        if (node->right) // Continue with the right child if present.
            playGame(node->right);
        else
            cout << "End of Binary Tree Trivia.\n\n"; // End if no right child.
    }
}

~BinaryTree() {
    deleteTree(root); // Clean up all nodes in the tree.
}

private:
// Helper function to delete the entire binary tree.
void deleteTree(Node* node) {
    if (node) {
        deleteTree(node->left); // Delete left subtree.
        deleteTree(node->right); // Delete right subtree.
        delete node; // Delete current node.
    }
}

};

//Binary Search Tree (BST) Implementation
class BST {

```

```

public:
struct Node {
string question; // Holds the trivia question.
string answer; // Holds the correct answer.
Node* left; // Pointer to the left child.
Node* right; // Pointer to the right child.
Node(string q, string a) : question(q), answer(a), left(nullptr), right(nullptr) {}
};
Node* root;
BST() : root(nullptr) {}
// Inserts a question and answer into the BST based on alphabetical order.
void insert(Node*& node, string question, string answer) {
if (!node) {
node = new Node(question, answer); // Create a new node if current node is null.
return;
}
if (question < node->question) // Insert in left subtree if question is smaller.
insert(node->left, question, answer);
else // Insert in right subtree otherwise.
insert(node->right, question, answer);
}
// Recursive function to play the trivia game using BST traversal.
void playGame(Node* node) {
if (!node) return; // Base case: If node is null, exit.
// Ask the current question.
cout << node->question << "\nYour Answer: ";
string userAnswer;
getline(cin, userAnswer);
transform(userAnswer.begin(), userAnswer.end(), userAnswer.begin(), ::tolower);
string correctAnswer = node->answer;
transform(correctAnswer.begin(), correctAnswer.end(), correctAnswer.begin(),
::tolower);
if (userAnswer == correctAnswer) {
cout << "Correct!\n\n";
if (node->left) // Continue with the left child if present.
playGame(node->left);
else
cout << "End of BST Trivia.\n\n"; // End if no left child.
} else {
cout << "Wrong! The correct answer was: " << node->answer << "\n\n";
}
}

```

```

if (node->right) // Continue with the right child if present.
playGame(node->right);
else
cout << "End of BST Trivia.\n\n"; // End if no right child.
}
}
~BST() {
deleteTree(root); // Clean up all nodes in the tree.
}
private:
// Helper function to delete the entire BST.
void deleteTree(Node* node) {
if (node) {
deleteTree(node->left); // Delete left subtree.
deleteTree(node->right); // Delete right subtree.
delete node; // Delete current node.
}
}
};
//Heap Implementation
class Heap {
private:
struct Trivia {
string question; // Holds the trivia question.
string answer; // Holds the correct answer.
int difficulty; // Holds the difficulty level (higher = harder).
Trivia(string q, string a, int d) : question(q), answer(a), difficulty(d) {}
};
vector<Trivia> heap; // The heap is stored as a vector.
// Helper function to maintain the max-heap property when inserting.
void heapifyUp(int index) {
int parent = (index - 1) / 2;
if (index && heap[index].difficulty > heap[parent].difficulty) {
swap(heap[index], heap[parent]); // Swap with parent if necessary.
heapifyUp(parent); // Recursively heapify up.
}
}
// Helper function to maintain the max-heap property when removing.
void heapifyDown(int index) {
int left = 2 * index + 1;

```

```

int right = 2 * index + 2;
int largest = index;
if (left < heap.size() && heap[left].difficulty > heap[largest].difficulty)
    largest = left;
if (right < heap.size() && heap[right].difficulty > heap[largest].difficulty)
    largest = right;
if (largest != index) {
    swap(heap[index], heap[largest]); // Swap with the largest child.
    heapifyDown(largest); // Recursively heapify down.
}
}
public:
// Inserts a question, answer, and difficulty level into the heap.
void insert(string question, string answer, int difficulty) {
    heap.push_back(Trivia(question, answer, difficulty)); // Add to the heap.
    heapifyUp(heap.size() - 1); // Restore heap property.
}
// Plays the trivia game, asking questions in order of difficulty.
void playGame() {
    while (!heap.empty()) {
        Trivia current = heap.front(); // Get the question with the highest difficulty.
        pop(); // Remove it from the heap.
        // Ask the current question.
        cout << "Q: " << current.question << "\nYour Answer: ";
        string userAnswer;
        getline(cin, userAnswer);
        transform(userAnswer.begin(), userAnswer.end(), userAnswer.begin(), ::tolower);
        string correctAnswer = current.answer;
        transform(correctAnswer.begin(), correctAnswer.end(), correctAnswer.begin(),
            ::tolower);
        if (userAnswer == correctAnswer) {
            cout << "Correct!\n\n";
        } else {
            cout << "Wrong! The correct answer was: " << current.answer << "\n\n";
        }
        if (heap.empty()) {
            cout << "End of Heap Trivia.\n\n";
        }
    }
}
}

```

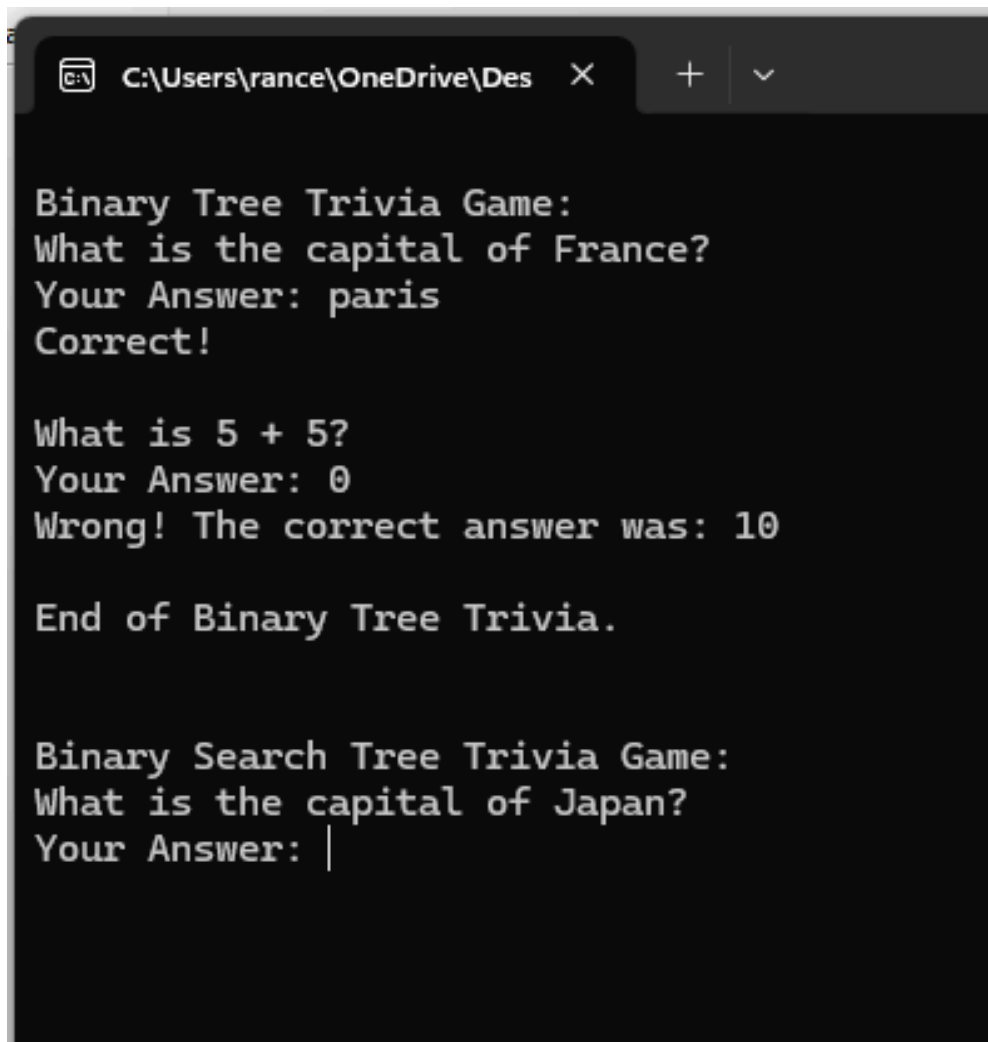
```

private:
// Removes the root of the heap and restores heap property.
void pop() {
if (heap.empty()) return;
heap[0] = heap.back(); // Move the last element to the root.
heap.pop_back(); // Remove the last element.
heapifyDown(0); // Restore heap property.
}
};
// ----- Main Function ----- //
int main() {
// Binary Tree Trivia
BinaryTree bt;

cout << "\nBinary Tree Trivia Game:" << endl;
bt.insert(bt.root, "What is the capital of France?", "Paris");
bt.insert(bt.root, "What is 5 + 5?", "10");
bt.insert(bt.root, "What is the largest ocean?", "Pacific");
bt.insert(bt.root, "Who discovered gravity?", "Newton");
bt.insert(bt.root, "What is the boiling point of water in Celsius?", "100");
bt.playGame(bt.root);
// Binary Search Tree Trivia
BST bst;
cout << "\nBinary Search Tree Trivia Game:" << endl;
bst.insert(bst.root, "What is the capital of Japan?", "Tokyo");
bst.insert(bst.root, "What is the tallest mountain?", "Everest");
bst.insert(bst.root, "Who wrote 'Hamlet'?", "Shakespeare");
bst.insert(bst.root, "What is the smallest planet in the Solar System?", "Mercury");
bst.insert(bst.root, "What is the freezing point of water in Fahrenheit?", "32");
bst.playGame(bst.root);
// Heap Trivia
Heap heap;
cout << "\nHeap Trivia Game (Difficulty-Based):" << endl;
heap.insert("What is the speed of light in m/s?", "299792458", 5);
heap.insert("What is the square root of 16?", "4", 1);
heap.insert("Who painted the Mona Lisa?", "Da Vinci", 3);
heap.insert("What is the chemical symbol for gold?", "Au", 4);
heap.insert("What is the largest planet in the Solar System?", "Jupiter", 2);
heap.playGame();
return 0;

```

}
V. OUTPUT



```
C:\Users\rance\OneDrive\Desktop > .\BinaryTreeTriviaGame.exe

Binary Tree Trivia Game:
What is the capital of France?
Your Answer: paris
Correct!

What is 5 + 5?
Your Answer: 0
Wrong! The correct answer was: 10

End of Binary Tree Trivia.

Binary Search Tree Trivia Game:
What is the capital of Japan?
Your Answer: |
```