ΠΑΝΕΠΙΣΤΗΜΙΟ ΠΕΙΡΑΙΩΣ

ΣΧΟΛΗ ΤΕΧΝΟΛΟΓΙΩΝ ΠΛΗΡΟΦΟΡΙΚΗΣ ΚΑΙ ΤΗΛΕΠΙΚΟΙΝΩΝΙΩΝ

ΤΜΗΜΑ ΠΛΗΡΟΦΟΡΙΚΗΣ

1η Εργασία Δομών Δεδομένων

ΜΑΤΙΝΑ ΠΑΠΑΔΑΚΟΥ Π21127

Βιβλιοθήκες

```
#include <iostream>
#include <algorithm>
#include <vector>
#include <random>
#include <string>
#include <limits>
using namespace std;
```

Αυτός ο κώδικας εισάγει διάφορες βασικές βιβλιοθήκες C++, συμπεριλαμβανομένων των **<iostream>, <algorithm>, <vector>, <random>, <string>** και **<limits>,** και χρησιμοποιεί το **namespace std** για να μην χρειάζεται να γράφουμε std:: πριν από κάθε χρήση βιβλιοθήκης.

<iostream>: Χρησιμοποιείται για την είσοδο και έξοδο δεδομένων.

<algorithm>: Περιέχει πολλούς χρήσιμους αλγορίθμους, όπως η ταξινόμηση.

<vector>: Χρησιμοποιείται για τη δημιουργία δυναμικών πινάκων.

<random>: Χρησιμοποιείται για τη δημιουργία τυχαίων αριθμών.

<string>: Χρησιμοποιείται για την εργασία με συμβολοσειρές.

<limits>: Περιέχει τις μέγιστες τιμές διαφόρων τύπων δεδομένων.

Κλάσεις για την Διπλά Συνδεδεμένη Λίστα Double και DoubleNode

```
template <class T>
class DoubleNode; // Forward declaration for DoubleNode
```

Αρχικά περιέχεται μία προκατασκευασμένη δήλωση (forward declaration) της κλάσης **DoubleNode**. Αυτό σημαίνει ότι ο μεταγλωττιστής γνωρίζει την ύπαρξη της κλάσης **DoubleNode**, αλλά δεν έχει ακόμη δει την υλοποίησή της.

Class Double

```
template <class T>
class Double {
private:
    DoubleNode<T>* LeftEnd, * RightEnd; // Pointers to the first and last nodes

public:
    Double() : LeftEnd(nullptr), RightEnd(nullptr) {} // Constructor for initializing pointers to nullptr
    ~Double(); // Destructor
    bool IsEmpty() const { return LeftEnd == nullptr; } // Function to check if the list is empty
    Double<T>& Insert(int k, const T& x); // Function to insert a node at a specific position
    Double<T>& Insert(const T& x); // Function to insert a node at the end
    void showData() const; // Function to display the data in the list
    void Delete(int k); // Function to delete a node at a specific position
    DoubleNode<T>* getNthNode(int n) const; // Function to get the nth node in the list
    int Size() const; // Function to get the size of the list
    void sortByAlphanumeric(string(T::* getFunc)() const); // Function to sort the list by alphanumeric order
    void sortByLikes(); // Function to sort the list by likes

    // Function to get a pointer to the LeftEnd node
    DoubleNode<T>* getFirstNode() const {
        return LeftEnd;
    }
};
```

Αυτός ο κώδικας υλοποιεί μια διπλά συνδεδεμένη λίστα (**Double Linked List**) με τη χρήση δύο διακριτών δεικτών **LeftEnd** και **RightEnd**, που δείχνουν στην αρχή και το τέλος της λίστας αντίστοιχα.

Σύντομη επεξήγηση των μεθόδων που περιέχει η κλάση:

IsEmpty(): Ελέγχει αν η λίστα είναι άδεια.

Insert(int k, const T& x): Εισάγει έναν κόμβο στη θέση k της λίστας με τιμή x.

Insert(const T& x): Εισάγει έναν κόμβο στο τέλος της λίστας με τιμή x.

showData(): Εμφανίζει τα δεδομένα που περιέχονται στους κόμβους της λίστας.

Delete(int k): Διαγράφει τον κόμβο στη θέση k της λίστας.

getNthNode(int n): Επιστρέφει έναν δείκτη στον n-οστό κόμβο της λίστας.

Size(): Επιστρέφει το μέγεθος της λίστας.

sortByAlphanumeric(string(T::* getFunc)() const): Ταξινομεί τη λίστα με βάση το όνομα το αντίστοιχου getter κάθε φορά που δίνεται ως συμβολοσειρά.

sortByLikes(): Ταξινομεί τη λίστα με βάση τον αριθμό των "likes".

getFirstNode(): επιστρέφει έναν δείκτη στον πρώτο κόμβο της λίστας.

Class DoubleNode

```
template <class T>
class DoubleNode {
    friend class Double<T>;
private:
    T data;
    DoubleNode<T>* left, * right;
public:
    DoubleNode() : left(nullptr), right(nullptr) {} // Default constructor
    // Constructor to initialize data and links
    DoubleNode(const T& _data) : data(_data), left(nullptr), right(nullptr) {}
    T getData() const { return data; } // Getter function for data
    DoubleNode<T>* getLink() const { return right; } // Use right pointer to get the link
    DoubleNode<T>* getLeft() const { return left; } // Getter function for left pointer
};
```

Αυτός ο κώδικας υλοποιεί τη δομή ενός κόμβου μιας διπλά συνδεδεμένης λίστα (**Doubly Linked List**).

Εδώ είναι μια σύντομη επεξήγηση των μελών της κλάσης **DoubleNode**:

data: Ένα αντικείμενο τύπου T που περιέχει τα δεδομένα του κόμβου.

left: Δείκτης που δείχνει στον προηγούμενο κόμβο στη λίστα.

right: Δείκτης που δείχνει στον επόμενο κόμβο στη λίστα.

Ο **default constructor** δημιουργεί έναν κόμβο χωρίς δεδομένα ή συνδέσμους.

Ο **constructor** που παίρνει ένα αντικείμενο τύπου T χρησιμοποιείται για να αρχικοποιήσει τα δεδομένα του κόμβου και τους συνδέσμους.

Οι μέθοδοι **getData()** και **getLink()** επιστρέφουν τα δεδομένα του κόμβου και τον επόμενο κόμβο αντίστοιχα.

Η μέθοδος **getLeft()** επιστρέφει τον προηγούμενο κόμβο στη λίστα.

Μέθοδοι για την Λειτουργία της Διπλά Συνδεδεμένης Λίστας

Μέθοδος sortByLikes()

```
template <class T>
void Double<T>::sortByLikes() {
    // Check if there are one or fewer elements
    if (LeftEnd == nullptr || LeftEnd->right == nullptr)
        return;

    bool swapped;
    DoubleNode<T>* p;
    DoubleNode<T>* q = nullptr;

    // Bubble sort algorithm
    do {
        swapped = false;
        p = LeftEnd;

        // Traverse the list
        while (p->right != q) {
            // If current node has fewer likes than its next node, swap them
            if (p->data.getLikes() < p->right->data.getLikes()) {
                // Swap nodes
                DoubleNode<T>* temp = p->right;
                p->right = temp->right;
                if (temp->right)
                    temp->right->left = p; // Update the left link of the new right node
                temp->left = p->left;
                if (p->left)
                    p->left->right = temp; // Update the right link of the new left node
                temp->right = p;
                p->left = temp;

                // Update LeftEnd if necessary
                if (p == LeftEnd)
                    LeftEnd = temp;

                swapped = true;
            }
            else {
                p = p->right;
            }
        }
        q = p;
    } while (swapped);
}
```

Αυτός ο κώδικας υλοποιεί τη μέθοδο **sortByLikes()** για την ταξινόμηση των στοιχείων μιας διπλά συνδεδεμένης λίστας βάσει του αριθμού των likes που έχει κάθε στοιχείο. Συγκεκριμένα, η μέθοδος ξεκινάει ελέγχοντας αν υπάρχει ένα ή λιγότερα στοιχεία στη λίστα. Αν ναι, τότε δεν χρειάζεται να γίνει ταξινόμηση. Χρησιμοποιείται η μέθοδος "bubble sort" για την ταξινόμηση των στοιχείων. Αυτή η μέθοδος είναι αποτελεσματική για μικρές λίστες. Ένας εξωτερικός βρόγχος επαναλαμβάνεται μέχρι να μην υπάρχουν άλλες ανταλλαγές στοιχείων στη λίστα. Αυτό εξασφαλίζει ότι όλα τα στοιχεία της λίστας είναι ταξινομημένα. Κάθε εσωτερικός βρόγχος περνά από όλα τα στοιχεία της λίστας και ανταλλάσσει

τα στοιχεία που έχουν περισσότερα likes από τα στοιχεία που έχουν λιγότερα. Οι ανταλλαγές γίνονται με την αλλαγή των δεικτών των κόμβων που αντιστοιχούν στα στοιχεία. Η διαδικασία επαναλαμβάνεται μέχρι να μην υπάρχουν άλλες ανταλλαγές στοιχείων.

Μέθοδος sortByAlphanumeric(string(T::* getFunc)() const)

```
template <class T>
void Double<T>::sortByAlphanumeric(string(T::* getFunc)() const) {
    // Check if there are one or fewer elements
    if (LeftEnd == nullptr || LeftEnd->right == nullptr)
        return;

    bool swapped;
    DoubleNode<T>* p;
    DoubleNode<T>* q = nullptr;

    // Bubble sort algorithm
    do {
        swapped = false;
        p = LeftEnd;

        // Traverse the list
        while (p->right != q) {
            // If current node's data retrieved by getFunc is greater than next node's data, swap them
            if ((p->data.*getFunc)() > (p->right->data.*getFunc)()) {
                // Swap nodes
                DoubleNode<T>* temp = p->right;
                p->right = temp->right;
                if (temp->right)
                    temp->right->left = p; // Update the left link of the new right node
                temp->left = p->left;
                if (p->left)
                    p->left->right = temp; // Update the right link of the new left node
                temp->right = p;
                p->left = temp;

                // Update LeftEnd if necessary
                if (p == LeftEnd)
                    LeftEnd = temp;

                swapped = true;
            }
            else {
                p = p->right;
            }
        }
        q = p;
    } while (swapped);
}
```

Αυτή η μέθοδος **sortByAlphanumeric()** υλοποιεί έναν αλγόριθμο ταξινόμησης που ταξινομεί τα στοιχεία μιας διπλά συνδεδεμένης λίστας βάσει του αντίστοιχου κάθε φορά Getter. Συγκεκριμένα, η μέθοδος ξεκινάει ελέγχοντας αν υπάρχει ένα ή λιγότερα στοιχεία στη λίστα. Αν ναι, τότε δεν χρειάζεται να γίνει ταξινόμηση. Χρησιμοποιείται η μέθοδος "bubble sort" για την ταξινόμηση των στοιχείων. Αυτή η μέθοδος είναι αποτελεσματική για μικρές λίστες. Ένας εξωτερικός βρόγχος επαναλαμβάνεται μέχρι να μην υπάρχουν άλλες ανταλλαγές στοιχείων στη λίστα. Αυτό εξασφαλίζει ότι όλα τα στοιχεία της λίστας είναι ταξινομημένα. Κάθε εσωτερικός βρόγχος περνά από όλα τα στοιχεία της λίστας και ανταλλάσσει τα στοιχεία που έχουν περισσότερα likes από τα στοιχεία που έχουν λιγότερα. Οι ανταλλαγές γίνονται

με την αλλαγή των δεικτών των κόμβων που αντιστοιχούν στα στοιχεία. Η διαδικασία επαναλαμβάνεται μέχρι να μην υπάρχουν άλλες ανταλλαγές στοιχείων.

Μέθοδος ~Double()

```
template <class T>
Double<T>::~Double() {
    // Start from LeftEnd and delete all nodes
    DoubleNode<T>* current = LeftEnd;
    while (current != nullptr) {
        DoubleNode<T>* next = current->right; // Traverse using the right pointer
        delete current;
        current = next;
    }
    LeftEnd = nullptr; // Set the LeftEnd pointer to nullptr after deleting all nodes
    RightEnd = nullptr; // Set the RightEnd pointer to nullptr after deleting all nodes
}
```

Αυτή η μέθοδος **~Double()** είναι ο καταστροφέας (destructor) για την κλάση Double. Σκοπός της είναι να διαγράψει όλους τους κόμβους της διπλά συνδεδεμένης λίστας και να απελευθερώσει τη μνήμη που καταλαμβάνουν αυτοί οι κόμβοι.

Η μέθοδος ξεκινά από τον κόμβο LeftEnd της λίστας και διαγράφει σταδιακά όλους τους κόμβους. Χρησιμοποιεί έναν δείκτη current για να περιηγηθεί στη λίστα, διαγράφοντας κάθε κόμβο που συναντάει. Κάθε φορά που διαγράφεται ένας κόμβος, ο δείκτης current μετακινείται στον επόμενο κόμβο χρησιμοποιώντας τον δεξιό δείκτη (right). Η διαδικασία επαναλαμβάνεται μέχρι να φτάσει σε έναν κόμβο που δείχνει σε nullptr, που σημαίνει ότι έχει φτάσει στο τέλος της λίστας. Τέλος, οι δείκτες LeftEnd και RightEnd ορίζονται ως nullptr για να δείξουν ότι η λίστα είναι τώρα κενή και η μνήμη έχει απελευθερωθεί.

Μέθοδος Size()

```
template <class T>
int Double<T>::Size() const {
    // Start from LeftEnd and count the number of nodes
    DoubleNode<T>* current = LeftEnd;
    int count = 0;
    while (current != nullptr) {
        count++;
        current = current->right; // Traverse using the right pointer
    }
    return count;
}
```

Αυτή η μέθοδος **Size()** επιστρέφει το μέγεθος της λίστας, δηλαδή τον αριθμό των κόμβων που περιέχει η διπλά συνδεδεμένη λίστα.

Ξεκινάει από τον αριστερό άκρο (LeftEnd) της λίστας και μετράει τους κόμβους περνώντας από τον έναν στον άλλο με τη χρήση του δεξιού δείκτη (right) του κάθε κόμβου. Όσο ο τρέχον κόμβος δεν είναι nullptr, αυξάνει τον μετρητή και προχωράει στον επόμενο κόμβο. Αφού περάσει από όλους τους κόμβους, επιστρέφει τον αριθμό που έχει μετρήσει, δηλαδή το μέγεθος της λίστας.

Μέθοδος getNthNode(int n)

```
template <class T>
DoubleNode<T>* Double<T>::getNthNode(int n) const {
    // Start from LeftEnd and traverse to the nth node
    DoubleNode<T>* current = LeftEnd;
    int count = 0;
    while (current != nullptr && count < n) {
        current = current->right; // Traverse using the right pointer
        count++;
    }
    return current; // Return the nth node or nullptr if not found
}
```

Αυτή η μέθοδος **getNthNode()** επιστρέφει τον κόμβο που βρίσκεται στην n-οστή θέση της λίστας. Ξεκινάει από την αρχή της λίστας και προχωρά στον επόμενο κόμβο χρησιμοποιώντας το δεξιό δείκτη μέχρι να φτάσει στην n-οστή θέση ή μέχρι να φτάσει στο τέλος της λίστας. Αν βρει τον κόμβο στην n-οστή θέση, επιστρέφει τον δείκτη, αλλιώς επιστρέφει nullptr.

Μέθοδος Delete(int k)

```
template <class T>
void Double<T>::Delete(int k) {
    if (k < 0) {
        cout << "Invalid index!" << endl; // Print error message if index is negative
        return;
    }

    if (k == 0) {
        if (LeftEnd == nullptr) {
            cout << "List is empty!" << endl; // Print error message if list is empty
            return;
        }

        // Delete the first node
        DoubleNode<T>* temp = LeftEnd;
        LeftEnd = LeftEnd->right;
        if (LeftEnd)
            LeftEnd->left = nullptr; // Update the left pointer of the new first node
        delete temp;
        return;
    }

    // Traverse to the (k-1)th node
    DoubleNode<T>* p = LeftEnd;
    for (int index = 1; index < k && p->right != nullptr; index++)
        p = p->right;

    if (p->right == nullptr) {
        cout << "Invalid index!" << endl; // Print error message if index is out of bounds
        return;
    }

    // Delete the kth node
    DoubleNode<T>* temp = p->right;
    p->right = temp->right;
    if (temp->right)
        temp->right->left = p; // Update the left pointer of the next node
    delete temp;
}
```

Αυτή η μέθοδος **Delete()** διαγράφει τον κόμβο στη θέση k από τη λίστα διπλής σύνδεσης. Αρχικά ελέγχει αν ο δείκτης k είναι αρνητικός ή αν η λίστα είναι άδεια. Αν ο δείκτης k είναι 0, τότε διαγράφει τον πρώτο κόμβο της λίστας. Διαφορετικά, προχωρά στον (k-1)ο κόμβο και διαγράφει τον κόμβο στη θέση k, ενώ ενημερώνει τους δείκτες αριστερά και δεξιά του κατάλληλα.

Μέθοδος Insert(int k, const T& x)

```
// Inserts an element at a specific position in the list. If the user tries to insert at a position greater than the list size, it is appended at the end.
template <class T>  <T> Provide sample template arguments for IntelliSense
Double<T>& Double<T>::Insert(int k, const T& x) {
    if (k < 0) {
        cout << "Invalid index!" << endl; // Print error message if index is negative
        return *this;
    }

    DoubleNode<T>* y = new DoubleNode<T>(x); // Create a new node with the provided data

    if (k == 0 || IsEmpty()) { // Insert at the beginning if list is empty or k is 0
        y->right = LeftEnd;
        if (LeftEnd) {
            LeftEnd->left = y; // Update the left link of the old first node
        }
        LeftEnd = y;
        if (!RightEnd) { // If RightEnd is nullptr, then this is the first node in the list
            RightEnd = y;
        }
        return *this;
    }

    DoubleNode<T>* p = LeftEnd;
    int index = 1;
    while (index < k && p->right) {
        p = p->right;
        index++;
    }

    //If k is bigger than the index of the list it adds the element at the end of the list
    // Insert the new node between p and p->right
    y->left = p;
    y->right = p->right;
    if (p->right) {
        p->right->left = y; // Update the left link of the node after the new node
    }
    p->right = y;

    if (p == RightEnd) { // Update RightEnd if the new node is inserted at the end or if k is bigger than the index
        RightEnd = y;
    }

    return *this;
}
```

Αυτή η μέθοδος **Insert()** εισάγει ένα στοιχείο σε μια συγκεκριμένη θέση στη διπλά συνδεδεμένη λίστα. Αν ο χρήστης προσπαθήσει να εισάγει σε μια θέση μεγαλύτερη από το μέγεθος της λίστας, το στοιχείο εισάγεται στο τέλος της λίστας.

Αρχικά ελέγχει αν ο δείκτης k είναι αρνητικός ή αν η λίστα είναι άδεια. Αν ο δείκτης k είναι 0 ή αν η λίστα είναι άδεια, τότε το νέο στοιχείο εισάγεται στην αρχή της λίστας. Διαφορετικά, η μέθοδος προχωρά στον ko κόμβο και εισάγει το νέο στοιχείο μεταξύ του p και του p->right, ενώ ενημερώνει τους δείκτες αριστερά και δεξιά του κατάλληλα. Επίσης, αν το k είναι μεγαλύτερο από το μέγεθος της λίστας, το νέο στοιχείο προστίθεται στο τέλος της λίστας.

Μέθοδος Insert(const T& x)

```
// Inserts an element at the end of the list
template <class T>
Double<T>& Double<T>::Insert(const T& x) {
    DoubleNode<T>* y = new DoubleNode<T>;
    y->data = x;
    y->left = nullptr;
    y->right = nullptr;

    if (LeftEnd == nullptr) { // If the list is empty
        LeftEnd = y;
    }
    else {
        DoubleNode<T>* p = LeftEnd;
        while (p->right != nullptr)
            p = p->right;

        p->right = y; // Insert new node at the end of the list
        y->left = p; // Update the left link of the new node
    }

    return *this;
}
```

Αυτή η μέθοδος **Insert()** εισάγει ένα στοιχείο στο τέλος της διπλά συνδεδεμένης λίστας. Αρχικά, δημιουργεί έναν νέο κόμβο με τα δεδομένα που δίνονται. Στη συνέχεια, ελέγχει αν η λίστα είναι άδεια. Αν είναι άδεια, ο νέος κόμβος γίνεται ο πρώτος κόμβος (LeftEnd). Διαφορετικά, προχωρά στον τελευταίο κόμβο της λίστας και εισάγει τον νέο κόμβο μετά από αυτόν. Στη συνέχεια, ενημερώνει τους δείκτες αριστερά και δεξιά του νέου κόμβου κατάλληλα.

Μέθοδος showData()

```
template <class T>
void Double<T>::showData() const {
    // Traverse the list and print data of each node
    DoubleNode<T>* current = LeftEnd;
    while (current != nullptr) {
        cout << "Title: " << current->data.getTitle() << endl;
        cout << "Artist's Name: " << current->data.getArtistsName() << endl;
        cout << "Artist's Surname: " << current->data.getArtistsSurname() << endl;
        cout << "Album Title: " << current->data.getAlbumTitle() << endl;
        cout << "Likes: " << current->data.getLikes() << endl;
        cout << "------------------------------" << endl;
        current = current->right; // Move to the next node using the right pointer
    }
    cout << endl;
}
```

Αυτή η μέθοδος **showData()** διασχίζει τη λίστα και εκτυπώνει τα δεδομένα κάθε κόμβου. Για κάθε κόμβο, εκτυπώνει τον τίτλο του τραγουδιού, το όνομα και το επίθετο του καλλιτέχνη, τον τίτλο του άλμπουμ και τον αριθμό των "likes". Έπειτα, προχωρά στον επόμενο κόμβο χρησιμοποιώντας τον δείκτη δεξιά. Τα δεδομένα κάθε κόμβου εκτυπώνονται ανάμεσα σε διαχωριστικές γραμμές για καλύτερη οργάνωση. Αυτός ο κώδικας βοηθάει στην προβολή των δεδομένων που αποθηκεύονται στους κόμβους της λίστας.

Η Κλάση Song

```cpp
class Song {
private:
    string title;
    string artists_name;
    string artists_surname;
    string album_title;
    int likes;

public:
    Song() : title(""), artists_name(""), artists_surname(""), album_title(""), likes(0) {} // Default constructor

    Song(const string& title, const string& artists_name, const string& artists_surname, const string& album_title, int likes)
        : title(title), artists_name(artists_name), artists_surname(artists_surname), album_title(album_title), likes(likes) {
    }

    string getTitle() const {
        return title;
    }

    string getArtistsName() const {
        return artists_name;
    }

    string getArtistsSurname() const {
        return artists_surname;
    }

    string getAlbumTitle() const {
        return album_title;
    }

    int getLikes() const {
        return likes;
    }
};
```

Αυτή η κλάση ονομάζεται **Song** και περιγράφει ένα τραγούδι. Έχει πέντε ιδιότητες: title, artists_name, artists_surname, album_title και likes, οι οποίες αντιπροσωπεύουν τον τίτλο του τραγουδιού, το όνομα του καλλιτέχνη, το επώνυμό του, τον τίτλο του άλμπουμ και τον αριθμό των "likes" αντίστοιχα.

Η κλάση παρέχει έναν default constructor που αρχικοποιεί τις μεταβλητές με κάποιες τιμές και έναν parameterized constructor που δέχεται τιμές για όλες τις ιδιότητες του τραγουδιού.

Επιπλέον, η κλάση παρέχει μεθόδους getTitle(), getArtistsName(), getArtistsSurname(), getAlbumTitle() και getLikes() που επιστρέφουν αντίστοιχα τον τίτλο του τραγουδιού, το όνομα και το επώνυμο του καλλιτέχνη, τον τίτλο του άλμπουμ και τον αριθμό των "likes". Αυτές οι μέθοδοι χρησιμοποιούνται για να αποκτηθεί πρόσβαση στις ιδιότητες του τραγουδιού από άλλα μέρη του προγράμματος.

Η Κλάση Playlist

```
class Playlist {
private:
    string playlists_name;
    Double <Song> songs;
    DoubleNode<Song>* current_song; // Pointer to the currently playing song
    int current_song_index = 0; // Initialize current song index
```

Η κλάση **Playlist** έχει τα ακόλουθα χαρακτηριστικά:

playlists_name: Ένα string που αποθηκεύει το όνομα της λίστας αναπαραγωγής.

songs: Ένα αντικείμενο της κλάσης Double<Song>, που αντιπροσωπεύει τα τραγούδια στη λίστα αναπαραγωγής.

current_song: Δείκτης προς το τρέχον τραγούδι που αναπαράγεται αυτήν την στιγμή.

current_song_index: Ένας ακέραιος που αποθηκεύει τον δείκτη του τρέχοντος τραγουδιού στη λίστα αναπαραγωγής. Αρχικοποιείται σε 0.

Αυτά τα χαρακτηριστικά βοηθούν στο να διαχειριστείτε και να αναπαράγετε τα τραγούδια στη λίστα αναπαραγωγής. Η λίστα αναπαραγωγής αποτελείται από ένα διπλά συνδεδεμένο λίστα τύπου Song, που περιέχει τα τραγούδια. Ο current_song δείχνει στο τρέχον τραγούδι που αναπαράγεται, ενώ ο current_song_index χρησιμοποιείται για να διατηρεί την τρέχουσα θέση στη λίστα.

Constructor

```
public:

    // Constructor to initialize playlist name
    Playlist(const string& playlists_name) : playlists_name(playlists_name) {};
```

Ο **constructor** της κλάσης Playlist δέχεται ένα όρισμα τύπου string που είναι το όνομα της λίστας αναπαραγωγής και το αποθηκεύει στο private member playlists_name. Αυτός ο κατασκευαστής χρησιμοποιείται για να δημιουργήσει ένα αντικείμενο τύπου Playlist με ένα συγκεκριμένο όνομα λίστας αναπαραγωγής.

Μέθοδος addSongToPlaylist(int position, const Song& newSong)

```
// Add a song to the playlist at a specific position
void addSongToPlaylist(int position, const Song& newSong) {
    songs.Insert(position, newSong);
}
```

Αυτή η μέθοδος προσθέτει ένα τραγούδι στη λίστα αναπαραγωγής σε μια συγκεκριμένη θέση. Παίρνει δύο ορίσματα: τη θέση όπου θα γίνει η εισαγωγή και το νέο τραγούδι που πρόκειται να προστεθεί.

Η λίστα αναπαραγωγής είναι μια διπλά συνδεδεμένη λίστα, οπότε η εισαγωγή του τραγουδιού στην κατάλληλη θέση γίνεται μέσω της μεθόδου **Insert** της κλάσης **Double**. Αν η θέση είναι εκτός ορίων της λίστας, το τραγούδι προστίθεται στο τέλος της λίστας.

Έτσι, αν η θέση είναι εντός ορίων, το τραγούδι εισάγεται στην καθορισμένη θέση, αλλιώς προστίθεται στο τέλος της λίστας. Μέσω αυτής της μεθόδου πραγματοποιείται το Ερώτημα 1.

Ερώτημα 1 εκτέλεση:

```
1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 1

Enter title of song : Blue Jeans
Enter name of singer : Lana
Enter surname of singer : Del Rey
Enter title of album : Born To Die
Enter number of likes : 1000
Enter position in playlist : 3
```

Μετά την εκτέλεση της εισαγωγής:

```
Playlist: Favourites
Songs:
Title: Bohemian Rhapsody
Artist's Name: Freddie
Artist's Surname: Mercury
Album Title: A Night at the Opera
Likes: 6842
------------------------------
Title: Social Cues
Artist's Name: Cage the
Artist's Surname: Elephant
Album Title: Social Cues
Likes: 5109
------------------------------
Title: Blue Jeans
Artist's Name: Lana
Artist's Surname: Del Rey
Album Title: Born To Die
Likes: 1000
------------------------------
Title: ZARI
Artist's Name: Marina
Artist's Surname: Satti
Album Title: ZARI
Likes: 3138
------------------------------
Title: Adiakopa
```

Μέθοδος addSongToPlaylist(const Song& newSong)

```
// Add a song to the end of the playlist
void addSongToPlaylist(const Song& newSong) {
    songs.Insert(newSong);
}
```

Αυτή η μέθοδος προσθέτει ένα τραγούδι στο τέλος της λίστας αναπαραγωγής. Χρησιμοποιεί τη μέθοδο Insert της κλάσης Double για να εισάγει το νέο τραγούδι στο τέλος της λίστας.

Συγκεκριμένα χρησιμοποιείται για την αρχικοποίηση του αντικειμένου playlist1.

Μέθοδος showPlaylistData()

```
// Display playlist data, including its name and all songs
void showPlaylistData() const {
    cout << "Playlist: " << playlists_name << endl;
    cout << "Songs:" << endl;
    songs.showData();
}
```

Αυτή η μέθοδος εμφανίζει τα δεδομένα της λίστας αναπαραγωγής, συμπεριλαμβανομένου του ονόματος της λίστας και όλων των τραγουδιών που περιέχει.

Πρώτα εκτυπώνει το όνομα της λίστας, και στη συνέχεια εμφανίζει τα δεδομένα κάθε τραγουδιού στη λίστα. Αυτό γίνεται χρησιμοποιώντας τη μέθοδο showData() της κλάσης Double, που εμφανίζει τα δεδομένα κάθε κόμβου στη διπλά συνδεδεμένη λίστα. Μέσω αυτής της μεθόδου πραγματοποιείται το Ερώτημα 7.

Ερώτημα 7 εκτέλεση:

```
1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 7

Playlist: Favourites
Songs:
Title: Bohemian Rhapsody
Artist's Name: Freddie
Artist's Surname: Mercury
Album Title: A Night at the Opera
Likes: 6842
------------------------------
Title: Social Cues
Artist's Name: Cage the
Artist's Surname: Elephant
Album Title: Social Cues
Likes: 5109
------------------------------
Title: ZARI
Artist's Name: Marina
Artist's Surname: Satti
Album Title: ZARI
Likes: 3138
------------------------------
Title: Adiakopa
Artist's Name: Foivos
Artist's Surname: Delivorias
Album Title: Exo
Likes: 2953
------------------------------
Title: Oloi Xoreuoun Me Thn K.
Artist's Name: Yorgos
Artist's Surname: Rous
Album Title: Rous
Likes: 1234
```

```
------------------------------
Title: The Yawning Grave
Artist's Name: Lord
Artist's Surname: Huron
Album Title: Strange Trails
Likes: 3478
------------------------------
Title: Riptide
Artist's Name: Vance
Artist's Surname: Joy
Album Title: Dream Your Life Away
Likes: 1042
------------------------------
Title: Girl, You'll Be a Woman Soon
Artist's Name: Urge
Artist's Surname: Overkill
Album Title: Stull - EP
Likes: 5527
------------------------------
Title: The Code
Artist's Name: Nemo
Artist's Surname: Melter
Album Title: The Code
Likes: 8042
------------------------------
```

Μέθοδος SortByAlphanumeric(string(T::* getFunc)() const)

```
// Sort the playlist by alphanumeric order based on a member function of Song
template <class T>
void SortByAlphanumeric(string(T::* getFunc)() const) {
    songs.sortByAlphanumeric(getFunc);
}
```

Αυτή η μέθοδος ταξινομεί τη λίστα τραγουδιών με βάση το αντίστοιχο αλφαριθμητικό στοιχείο που επιθυμεί ο χρήστης. Μέσω αυτής της μεθόδου πραγματοποιείται το Ερώτημα 6.

Ερώτημα 6 εκτέλεση 1. Sort playlist by song title:

```
Playlist: Favourites
Songs:
Title: Adiakopa
Artist's Name: Foivos
Artist's Surname: Delivorias
Album Title: Exo
Likes: 2953
------------------------------
Title: Bohemian Rhapsody
Artist's Name: Freddie
Artist's Surname: Mercury
Album Title: A Night at the Opera
Likes: 6842
------------------------------
Title: Girl, You'll Be a Woman Soon
Artist's Name: Urge
Artist's Surname: Overkill
Album Title: Stull - EP
Likes: 5527
------------------------------
Title: Oloi Xoreuoun Me Thn K.
Artist's Name: Yorgos
Artist's Surname: Rous
Album Title: Rous
Likes: 1234
------------------------------
Title: Riptide
Artist's Name: Vance
Artist's Surname: Joy
Album Title: Dream Your Life Away
Likes: 1042
------------------------------
Title: Social Cues
Artist's Name: Cage the
Artist's Surname: Elephant
Album Title: Social Cues
Likes: 5109
------------------------------
Title: The Code
Artist's Name: Nemo
Artist's Surname: Melter
Album Title: The Code
Likes: 8042
------------------------------
Title: The Yawning Grave
Artist's Name: Lord
Artist's Surname: Huron
Album Title: Strange Trails
Likes: 3478
------------------------------
```

```
What type of sorting do you want to do?
1. Sort playlist by song title
2. Sort playlist by singer's first name
3. Sort playlist by singer's last name
4. Sort playlist by number of likes (decreasing)
5. Cancel
Choose an option between 1-5: 1
The list has been sorted by title, press 7 to view the new list.
```

```
Title: ZARI
Artist's Name: Marina
Artist's Surname: Satti
Album Title: ZARI
Likes: 3138
--------------------------------
```

Ερώτημα 6 εκτέλεση 2. Sort playlist by singer's first name:

```
Playlist: Favourites
Songs:
Title: Social Cues
Artist's Name: Cage the
Artist's Surname: Elephant
Album Title: Social Cues
Likes: 5109
--------------------------------
Title: Adiakopa
Artist's Name: Foivos
Artist's Surname: Delivorias
Album Title: Exo
Likes: 2953
--------------------------------
Title: Bohemian Rhapsody
Artist's Name: Freddie
Artist's Surname: Mercury
Album Title: A Night at the Opera
Likes: 6842
--------------------------------
Title: The Yawning Grave
Artist's Name: Lord
Artist's Surname: Huron
Album Title: Strange Trails
Likes: 3478
--------------------------------
Title: ZARI
Artist's Name: Marina
Artist's Surname: Satti
Album Title: ZARI
Likes: 3138
--------------------------------
Title: The Code
Artist's Name: Nemo
Artist's Surname: Melter
Album Title: The Code
Likes: 8042
--------------------------------
Title: Girl, You'll Be a Woman Soon
Artist's Name: Urge
Artist's Surname: Overkill
Album Title: Stull - EP
Likes: 5527
--------------------------------
Title: Riptide
Artist's Name: Vance
Artist's Surname: Joy
Album Title: Dream Your Life Away
Likes: 1042
--------------------------------
```

```
What type of sorting do you want to do?
1. Sort playlist by song title
2. Sort playlist by singer's first name
3. Sort playlist by singer's last name
4. Sort playlist by number of likes (decreasing)
5. Cancel
Choose an option between 1-5: 2
The list has been sorted by the singer's first name, press 7 to view the new list.
```

```
Title: Oloi Xoreuoun Me Thn K.
Artist's Name: Yorgos
Artist's Surname: Rous
Album Title: Rous
Likes: 1234
--------------------------------
```

Ερώτημα 6 εκτέλεση 3. Sort playlist by singer's last name:

```
Playlist: Favourites
Songs:
Title: Adiakopa
Artist's Name: Foivos
Artist's Surname: Delivorias
Album Title: Exo
Likes: 2953
--------------------------------
Title: Social Cues
Artist's Name: Cage the
Artist's Surname: Elephant
Album Title: Social Cues
Likes: 5109
--------------------------------
Title: The Yawning Grave
Artist's Name: Lord
Artist's Surname: Huron
Album Title: Strange Trails
Likes: 3478
--------------------------------
Title: Riptide
Artist's Name: Vance
Artist's Surname: Joy
Album Title: Dream Your Life Away
Likes: 1042
--------------------------------
Title: The Code
Artist's Name: Nemo
Artist's Surname: Melter
Album Title: The Code
Likes: 8042
--------------------------------
Title: Bohemian Rhapsody
Artist's Name: Freddie
Artist's Surname: Mercury
Album Title: A Night at the Opera
Likes: 6842
--------------------------------
Title: Girl, You'll Be a Woman Soon
Artist's Name: Urge
Artist's Surname: Overkill
Album Title: Stull - EP
Likes: 5527
--------------------------------
Title: Oloi Xoreuoun Me Thn K.
Artist's Name: Yorgos
Artist's Surname: Rous
Album Title: Rous
Likes: 1234
--------------------------------
Title: ZARI
Artist's Name: Marina
Artist's Surname: Satti
Album Title: ZARI
Likes: 3138
--------------------------------
```

```
What type of sorting do you want to do?
1. Sort playlist by song title
2. Sort playlist by singer's first name
3. Sort playlist by singer's last name
4. Sort playlist by number of likes (decreasing)
5. Cancel
Choose an option between 1-5: 3
The list has been sorted by the singer's last name, press 7 to view the new list.
```

Ερώτημα 6 εκτέλεση 5. Cancel:

```
What type of sorting do you want to do?
1. Sort playlist by song title
2. Sort playlist by singer's first name
3. Sort playlist by singer's last name
4. Sort playlist by number of likes (decreasing)
5. Cancel
Choose an option between 1-5: 5

1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number:
```

Μέθοδος void SortByLikes()

```
// Sort the playlist by number of likes
void SortByLikes() {
    songs.sortByLikes();
}
```

Αυτή η μέθοδος ταξινομεί τη λίστα τραγουδιών βάσει του αριθμού των "likes" που έχουν τα τραγούδια. Μέσω αυτής της μεθόδου πραγματοποιείται το Ερώτημα 6.

Ερώτημα 6 εκτέλεση 4. Sort playlist by number of likes (decreasing):

```
Playlist: Favourites
Songs:
Title: The Code
Artist's Name: Nomo
Artist's Surname: Melter
Album Title: The Code
Likes: 8042
---------------------------------
Title: Bohemian Rhapsody
Artist's Name: Freddie
Artist's Surname: Mercury
Album Title: A Night at the Opera
Likes: 6842
---------------------------------
Title: Girl, You'll Be a Woman Soon
Artist's Name: Urge
Artist's Surname: Overkill
Album Title: Stull - EP
Likes: 5527
---------------------------------
Title: Social Cues
Artist's Name: Cage the
Artist's Surname: Elephant
Album Title: Social Cues
Likes: 5109
---------------------------------
Title: The Yawning Grave
Artist's Name: Lord
Artist's Surname: Huron
Album Title: Strange Trails
Likes: 3478
---------------------------------
Title: ZARI
Artist's Name: Marina
Artist's Surname: Satti
Album Title: ZARI
Likes: 3138
---------------------------------
Title: Adiakopa
Artist's Name: Foivos
Artist's Surname: Delivorias
Album Title: Exo
Likes: 2953
---------------------------------
Title: Oloi Xoreuoun Me Thn K.
Artist's Name: Yorgos
Artist's Surname: Rous
Album Title: Rous
Likes: 1234
---------------------------------
Title: Riptide
Artist's Name: Vance
Artist's Surname: Joy
Album Title: Dream Your Life Away
Likes: 1042
---------------------------------
```

```
What type of sorting do you want to do?
1. Sort playlist by song title
2. Sort playlist by singer's first name
3. Sort playlist by singer's last name
4. Sort playlist by number of likes (decreasing)
5. Cancel
Choose an option between 1-5: 4
The list has been sorted by how many people liked the video, press 7 to view the new list.
```

Μέθοδος playNextSong()

```
// Play the next song in the playlist
void playNextSong() {
    if (songs.IsEmpty()) {
        cout << "No songs in the playlist!" << endl;
        return;
    }

    if (current_song == nullptr || current_song->getLink() == nullptr) {
        // If there's no current song playing or if the current song is the last one,
        // reset current_song to the first song
        current_song = songs.getFirstNode();
        current_song_index = 0; // Reset current song index
    }
    else {
        // Move to the next song
        current_song = current_song->getLink();
        current_song_index++; // Increment current song index
    }

    // Display information about the currently playing song
    cout << "Playing song \"" << current_song->getData().getTitle() << "\" from the album \""
        << current_song->getData().getAlbumTitle() << "\" with singer \""
        << current_song->getData().getArtistsName() << " "
        << current_song->getData().getArtistsSurname() << "\" ("
        << current_song->getData().getLikes() << " users like this song)" << endl;
}
```

Αυτή η μέθοδος αναπαράγει το επόμενο τραγούδι στη λίστα αναπαραγωγής. Εάν δεν υπάρχουν τραγούδια στη λίστα, εκτυπώνει ένα μήνυμα σφάλματος. Αν δεν υπάρχει τρέχον τραγούδι που παίζεται ή αν το τρέχον τραγούδι είναι το τελευταίο, τότε το τρέχον τραγούδι επιστρέφεται στο πρώτο τραγούδι της λίστας αναπαραγωγής. Στη συνέχεια, εμφανίζονται πληροφορίες για το τρέχον τραγούδι που παίζεται, όπως ο τίτλος του, ο τίτλος του άλμπουμ, ο καλλιτέχνης και ο αριθμός των "likes" που έχει. Μέσω αυτής της μεθόδου πραγματοποιείται το Ερώτημα 2.

Ερώτημα 2 εκτέλεση:

```
1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 2

Playing song "Bohemian Rhapsody" from the album "A Night at the Opera" with singer "Freddie Mercury" (6842 users like this song)

1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 2

Playing song "Social Cues" from the album "Social Cues" with singer "Cage the Elephant" (5109 users like this song)

1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 2

Playing song "ZARI" from the album "ZARI" with singer "Marina Satti" (3138 users like this song)

1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
```

Μέθοδος playPreviousSong()

```
// Play the previous song in the playlist
void playPreviousSong() {
    if (songs.IsEmpty()) {
        cout << "No songs in the playlist!" << endl;
        return;
    }

    // If there's no current song playing, set current_song to the first song
    if (current_song == nullptr) {
        cout << "No song is currently playing." << endl;
        return;
    }

    // Get the previous song
    DoubleNode<Song>* prev_song = current_song->getLeft();

    // If there is no previous song, i.e., current song is the first song in the playlist
    if (prev_song == nullptr) {
        cout << "No previous song. You are already at the beginning of the playlist playing the 1st song." << endl;
        return;
    }

    // Set current song to the previous song
    current_song = prev_song;

    // Decrement current song index
    current_song_index--;

    // Display information about the currently playing song
    cout << "Playing song \"" << current_song->getData().getTitle() << "\" from the album \""
        << current_song->getData().getAlbumTitle() << "\" with singer \""
        << current_song->getData().getArtistsName() << " "
        << current_song->getData().getArtistsSurname() << "\" ("
        << current_song->getData().getLikes() << " users like this song)" << endl;
}
```

Αυτή η μέθοδος αναπαράγει το προηγούμενο τραγούδι στη λίστα αναπαραγωγής. Ελέγχει αν η λίστα είναι άδεια. Εάν είναι, εμφανίζει ένα μήνυμα λάθους και τερματίζει τη μέθοδο. Αν δεν υπάρχει τρέχον τραγούδι, τοποθετεί το δείκτη current_song στο πρώτο τραγούδι της λίστας. Επιπλέον αναζητά το προηγούμενο τραγούδι από το τρέχον τραγούδι. Αν δεν υπάρχει προηγούμενο τραγούδι, εμφανίζει ένα μήνυμα που δηλώνει ότι ο χρήστης βρίσκεται ήδη στην αρχή της λίστας και τερματίζει τη μέθοδο. Επιπλέον τοποθετεί τον δείκτη current_song στο προηγούμενο τραγούδι και μειώνει τον δείκτη current_song_index κατά ένα. Τέλος εμφανίζει πληροφορίες για το τρέχον τραγούδι που αναπαράγεται. Μέσω αυτής της μεθόδου πραγματοποιείται το Ερώτημα 3.

Ερώτημα 3 εκτέλεση:

```
1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 3

Playing song "Social Cues" from the album "Social Cues" with singer "Cage the Elephant" (5109 users like this song)

1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 3

Playing song "Bohemian Rhapsody" from the album "A Night at the Opera" with singer "Freddie Mercury" (6842 users like this song)

1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 3

No previous song. You are already at the beginning of the playlist playing the 1st song.
```

Μέθοδος showAllSongsRandomly()

```
// Display all songs in the playlist in random order
void showAllSongsRandomly() {
    if (songs.IsEmpty()) {
        cout << "No songs in the playlist!" << endl;
        return;
    }

    // Create a vector of song pointers
    vector<DoubleNode<Song>*> songPointers;
    DoubleNode<Song>* current = songs.getFirstNode();
    while (current != nullptr) {
        songPointers.push_back(current);
        current = current->getLink();
    }

    // Shuffle the vector
    random_device rd;
    mt19937 rng(rd());
    shuffle(songPointers.begin(), songPointers.end(), rng);

    // Display the songs in random order
    for (DoubleNode<Song>* songNode : songPointers) {
        cout << "Playing song \"" << songNode->getData().getTitle() << "\" from the album \""
            << songNode->getData().getAlbumTitle() << "\" with singer \""
            << songNode->getData().getArtistsName() << " "
            << songNode->getData().getArtistsSurname() << "\" ("
            << songNode->getData().getLikes() << " users like this song)" << endl;
    }
}
```

Αυτή η μέθοδος αναπαράγει όλα τα τραγούδια στη λίστα αναπαραγωγής με τυχαία σειρά.

1. Ελέγχει αν η λίστα είναι άδεια. Αν είναι, εμφανίζει ένα μήνυμα λάθους και τερματίζει τη μέθοδο.
2. Δημιουργεί ένα vector δεικτών σε κάθε κόμβο της λίστας.
3. Ανακατεύει τα στοιχεία του vector με τη χρήση της συνάρτησης shuffle της STL.
4. Αναπαράγει τα τραγούδια στο vector σε τυχαία σειρά, εμφανίζοντας πληροφορίες για κάθε τραγούδι.

Αυτός ο τρόπος αναπαραγωγής των τραγουδιών σε τυχαία σειρά δίνει μια διαφορετική εμπειρία ακρόασης σε κάθε εκτέλεση της μεθόδου. Μέσω αυτής της μεθόδου πραγματοποιείται το Ερώτημα 4.

Ερώτημα 4 εκτέλεση:

```
1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 4

Playing song "The Yawning Grave" from the album "Strange Trails" with singer "Lord Huron" (3478 users like this song)
Playing song "Adiakopa" from the album "Exo" with singer "Foivos Delivorias" (2953 users like this song)
Playing song "The Code" from the album "The Code" with singer "Nemo Melter" (8042 users like this song)
Playing song "ZARI" from the album "ZARI" with singer "Marina Satti" (3138 users like this song)
Playing song "Social Cues" from the album "Social Cues" with singer "Cage the Elephant" (5109 users like this song)
Playing song "Oloi Xoreuoun Me Thn K." from the album "Rous" with singer "Yorgos Rous" (1234 users like this song)
Playing song "Bohemian Rhapsody" from the album "A Night at the Opera" with singer "Freddie Mercury" (6842 users like this song)
Playing song "Girl, You'll Be a Woman Soon" from the album "Stull - EP" with singer "Urge Overkill" (5527 users like this song)
Playing song "Riptide" from the album "Dream Your Life Away" with singer "Vance Joy" (1042 users like this song)
```

Μέθοδος transferSong(int fromPosition, int toPosition)

```
void transferSong(int fromPosition, int toPosition) {
    // Check if fromPosition and toPosition are valid
    if (fromPosition < 0 || fromPosition > songs.Size() || toPosition < 0 || toPosition > songs.Size()) {
        cout << "Invalid positions! Positions should be between 1 and " << songs.Size() << endl;
        return;
    }

    // Get the node at the fromPosition
    DoubleNode<Song>* fromNode = songs.getNthNode(fromPosition);
    if (!fromNode) {
        cout << "Invalid from position!" << endl;
        return;
    }

    // Get the node at the toPosition
    DoubleNode<Song>* toNode = songs.getNthNode(toPosition);
    if (!toNode) {
        cout << "Invalid to position!" << endl;
        return;
    }

    // Remove the song from the fromPosition
    Song songToTransfer = fromNode->getData();
    songs.Delete(fromPosition);

    // Insert the song at the toPosition
    songs.Insert(toPosition, songToTransfer);

    // Inform the user about the transfer
    cout << "Song '" << songToTransfer.getTitle() << "' transferred to position " << toPosition + 1 << endl;
}
```

Αυτή η συνάρτηση, transferSong, μεταφέρει ένα τραγούδι από μια θέση σε μια άλλη εντός της λίστας αναπαραγωγής. Συγκεκριμένα:

1. Ελέγχει αν οι θέσεις fromPosition και toPosition είναι έγκυρες, δηλαδή αν βρίσκονται εντός των ορίων της λίστας.
2. Λαμβάνει τον κόμβο στη θέση fromPosition.
3. Λαμβάνει τον κόμβο στη θέση toPosition.
4. Αφαιρεί το τραγούδι από τη θέση fromPosition.

5. Εισάγει το τραγούδι στη θέση toPosition.
6. Ενημερώνει το χρήστη για τη μεταφορά του τραγουδιού.

Μέσω αυτής της μεθόδου πραγματοποιείται το Ερώτημα 5.

Ερώτημα 5 εκτέλεση:

```
1. Insert new song into playlist
2. Play next song in playlist
3. Play previous song in playlist
4. Play all songs in playlist in random order
5. Move song to another position in playlist
6. Sort playlist
7. Print playlist to screen
8. Exit
Choose a number: 5

Enter position of song in playlist to move: 1
Enter new position of song in playlist :4
Song 'Bohemian Rhapsody' transferred to position 4
```

```
Playlist: Favourites
Songs:
Title: Social Cues
Artist's Name: Cage the
Artist's Surname: Elephant
Album Title: Social Cues
Likes: 5109
------------------------------
Title: ZARI
Artist's Name: Marina
Artist's Surname: Satti
Album Title: ZARI
Likes: 3138
------------------------------
Title: Adiakopa
Artist's Name: Foivos
Artist's Surname: Delivorias
Album Title: Exo
Likes: 2953
------------------------------
Title: Bohemian Rhapsody
Artist's Name: Freddie
Artist's Surname: Mercury
Album Title: A Night at the Opera
Likes: 6842
------------------------------
Title: Oloi Xoreuoun Me Thn K.
Artist's Name: Yorgos
Artist's Surname: Rous
Album Title: Rous
Likes: 1234
------------------------------
Title: The Yawning Grave
Artist's Name: Lord
Artist's Surname: Huron
Album Title: Strange Trails
Likes: 3478
------------------------------
```

Η Συνάρτηση InitializePlaylist(Playlist& playlist)

```
void static InitializePlaylist(Playlist& playlist)
{
    Song song1("Bohemian Rhapsody", "Freddie", "Mercury", "A Night at the Opera", 6842);
    Song song2("Social Cues", "Cage the", "Elephant", "Social Cues", 5109);
    Song song3("ZARI", "Marina", "Satti", "ZARI", 3138);
    Song song4("Adiakopa", "Foivos", "Delivorias", "Exo", 2953);
    Song song5("Oloi Xoreuoun Me Thn K.", "Yorgos", "Rous", "Rous", 1234);
    Song song6("The Yawning Grave", "Lord", "Huron", "Strange Trails", 3478);
    Song song7("Riptide", "Vance", "Joy", "Dream Your Life Away", 1042);
    Song song8("Girl, You'll Be a Woman Soon", "Urge", "Overkill", "Stull - EP", 5527);
    Song song9("The Code", "Nemo", "Melter", "The Code", 8042);

    playlist.addSongToPlaylist(song1);
    playlist.addSongToPlaylist(song2);
    playlist.addSongToPlaylist(song3);
    playlist.addSongToPlaylist(song4);
    playlist.addSongToPlaylist(song5);
    playlist.addSongToPlaylist(song6);
    playlist.addSongToPlaylist(song7);
    playlist.addSongToPlaylist(song8);
    playlist.addSongToPlaylist(song9);
}
```

Η συνάρτηση **InitializePlaylist** αρχικοποιεί τη λίστα αναπαραγωγής με ορισμένα τραγούδια. Κάθε τραγούδι δημιουργείται και προστίθεται στη λίστα. Ας δούμε πώς λειτουργεί:

Δημιουργούνται διάφορα αντικείμενα τύπου Song με σκοπό να αρχικοποιήσουν τη λίστα αναπαραγωγής. Κάθε τραγούδι έχει έναν τίτλο, το όνομα και το επίθετο του καλλιτέχνη, τον τίτλο του άλμπουμ και τον αριθμό των likes.

Τα τραγούδια προστίθενται στη λίστα αναπαραγωγής μέσω της κλήσης της συνάρτησης addSongToPlaylist.

Αυτή η διαδικασία εξασφαλίζει ότι η λίστα αναπαραγωγής αρχικοποιείται με έτοιμα τραγούδια, έτοιμα για αναπαραγωγή.

Η Συνάρτηση main()

Ο κώδικας της συνάρτησης **main()** αντιπροσωπεύει τον κύριο κώδικα ενός προγράμματος που διαχειρίζεται μια λίστα αναπαραγωγής τραγουδιών. Η λειτουργία της είναι:

1. Αρχικά, δημιουργείται ένα αντικείμενο της κλάσης Playlist με όνομα "Favourites".
2. Στη συνέχεια, καλείται η συνάρτηση InitializePlaylist, η οποία προετοιμάζει τη λίστα αναπαραγωγής.
3. Ακολουθεί ένας βρόχος για το κύριο μενού του προγράμματος, όπου ο χρήστης μπορεί να επιλέξει διάφορες επιλογές, όπως την εισαγωγή νέου τραγουδιού, την αναπαραγωγή του επόμενου ή προηγούμενου τραγουδιού, την αναπαραγωγή όλων των τραγουδιών σε τυχαία σειρά, τη μετακίνηση ενός τραγουδιού σε μια άλλη θέση στη λίστα, την ταξινόμηση της λίστας και την εκτύπωση της λίστας.
4. Κάθε ενέργεια από τον χρήστη εκτελείται μετά από έλεγχο εγκυρότητας της εισόδου.
5. Ο κύριος βρόχος επαναλαμβάνεται μέχρι ο χρήστης να επιλέξει την επιλογή "Exit".

Κάθε λειτουργία του μενού υλοποιείται μέσω κλήσεων σχετικών με το αντικείμενο της Playlist, όπως η προσθήκη νέου τραγουδιού, η αναπαραγωγή τραγουδιού, η ταξινόμηση της λίστας, κ.λπ. Η λειτουργία κάθε μενού επεξηγείται κατάλληλα στην οθόνη για τον χρήστη.