CT-159

DATA STRUCTURES ALGORITHMS AND APPLICATIONS

PROJECT REPORT

STRESS RELIEVER

GROUP MEMBERS:

- **ALEESHBA (CT-22005)**
- **YASHA ALI (CT-22010)**
- **BAZIGHA FAROOQ (CT-22021)**

COURSE INSTRUCTORS: **DR. USMAN AMJAD**

LAB INSTRUCTORS: **DR. KARIM KAZI**

TABLE OF CONTENTS

1. INTRODUCTION :

In our fast-paced modern world, stress has become an unwelcome companion for many, affecting individuals of all ages and backgrounds. The constant demands of daily life, work pressures, and personal challenges create a significant burden on our mental well-being. As we navigate through these stressors, seeking effective ways to unwind and find moments of tranquility becomes crucial. Enter the realm of gaming – a dynamic avenue that transcends mere entertainment, offering a unique and engaging escape from the pressures of everyday life. Over the years, video games, in particular, have evolved beyond mere amusement, demonstrating their potential as powerful tools for stress relief and relaxation. This project endeavors to harness the therapeutic potential of gaming to positively impact mental health.

2. PROBLEM STATEMENT:

Our stress reliever application, featuring “Melody craft”, “pen thought”, “flow free”, “questionnaire”, “calm canvas”, and “pop it”, aims to contribute significantly to mental health. The challenge lies in seamlessly integrating data structures like trees, stack, queue, arrays, and linked lists to optimize gameplay. The emphasis is on leveraging these structures not only to diversify stress-relieving activities but also to enhance the overall user experience. This initiative is dedicated to promoting mental health and well-being through an innovative approach, acknowledging the positive impact of immersive and enjoyable stress relief games.

3. UTILISING DATA STRUCTURES AND ALGORITHMS:

Balloon Game:

Linked List:

The Balloon Game utilizes a singly linked list data structure to manage the dynamic creation and destruction of balloons. The Balloon structure serves as the node in the linked

list, where each node represents an individual balloon. The linked list facilitates efficient traversal, removal of popped balloons, and dynamic memory management, allowing balloons to be added or removed during gameplay.

Piano

Linked Lists:

SINGLY LINKED LIST: Two instances of singly linked lists are used to manage the initial X and Y positions of the piano keys. These linked lists facilitate the dynamic positioning of piano keys, offering flexibility for potential adjustments in their arrangement during runtime.

CIRCULAR LINKED LIST: The code incorporates a Circular Linked List to manage a sequence of SDL textures representing images. This circular linked list allows seamless cycling through a collection of images, creating a continuous background effect that simulates a video playback.

Flow Free

Array

The program uses **2D arrays** to represent the game grid. These arrays store the colors of the dots and the visited status during the solving process. They efficiently manage the game state and track the connections between dots. Another **1D array** is also used to store RGBA color values for different dots in the FlowFree game.

A structure is employed to represent the RGBA color values. It simplifies the storage and retrieval of color information for each dot on the game grid.

Backtracking Algorithm implementation:

The backtracking algorithm employed in Flow Free enhances the solving process, intelligently navigating through the game grid to connect colored dots efficiently and contribute to a challenging yet enjoyable gaming experience.

Paint:

Stack

The **Stack** data structure, implemented using singly linked list is used for managing the undo and redo functionality in the Paint App. The stack is used to store the canvas state which keeps track of the canvas state at different points in time, allowing the user to undo and redo their actions.

Queue

The **Queue** data structure, made through singly linked list is employed to implement the flood-fill algorithm used for the bucket tool in the Paint App. It helps in efficiently processing adjacent pixels while filling the area with the selected color. A queue of pairs of integers representing pixel coordinates is used to store the pixels that need to be processed during the flood-fill operation. Several custom structures types are used for organizing and storing information related to buttons, points, and colored squares in the Paint App.

JOURNALING:

DoublyLinkedList is employed for managing journal entries, allowing for easy addition, deletion and viewing of notes, providing efficient traversal. The Date and Time of Notes are recorded and the entries are sorted accordingly. The DoublyLinkedList implementation in the code offers seamless management of journal entries with its bidirectional traversal, supporting efficient addition, deletion, and traversal operations. This enhances the user experience in the journaling stress reliever game.

QUESTIONNAIRE:

Binary Tree

The implementation of a questionnaire-driven stress reliever leverages fundamental data structures. The central data structure is the TreeNode class, forming a tree-like structure to represent the branching questions and responses. Each TreeNode encapsulates a question and two pointers, trueOption and falseOption, leading to subsequent nodes based on user responses. This hierarchical arrangement efficiently organizes the decision-making flow of the questionnaire. The code seamlessly integrates Object-Oriented Programming principles with data structures to create an interactive and engaging questionnaire experience, contributing to the overall effectiveness of the stress-relieving application.

OOP PILLARS UTILISATION:

- Abstraction: We have designed an abstract class which contains the data members and pure virtual methods common to all classes. The pure virtual methods are:
    - initialize () which is responsible for initializing the SDL components,
    - run () which is responsible for the proper running of the components, and
    - handleEvents() which handles different controls depending on the component.
- Inheritance: we have used two types of inheritance in our code:
    1. HIERARCHIAL INHERITANCE: The classes namely, “flowfree” , “paint” , “journaling” , “balloon pop” , “piano”, and “questionnaire” are derived from the abstract class.
- Encapsulation: For encapsulation, we converted the source files of the all six components to header files and included them in the Menu’s source file. We also declared the methods and data members of our component classes as private and protected. This ensured that our implementation is hidden from the user.

4. FEATURES OF OUR PROGRAM:

Some features of our program that distinguish ours from others are:

1. Our stress reliever project has visually stunning graphics that encompass a rich array of audio elements, intricate images, and various font styles.

2. Our program incorporates an interactive decision-tree-based questionnaire about the user's day. Users answer a series of yes/no questions, leading them through a dynamic path that culminates in personalized mental health suggestions based on their answers. This unique approach adds a fun and engaging element, providing users with customized recommendations based on their responses.
3. Our system offers a multitude of stress-relieving activities, each equipped with a unique set of features designed to captivate and engage users.
4. Our system features offer user-friendly navigation, empowering users to effortlessly comprehend the mechanics. This intuitive support gives a relaxing and stress-free experience, enabling them to excel and fully enjoy their stress-relieving endeavour.
5. Our program enables seamless transitions between different activities, providing users the freedom to explore various options without exiting the main screen. This maximizes their convenience and enjoyment by eliminating disruptions and inconveniences.

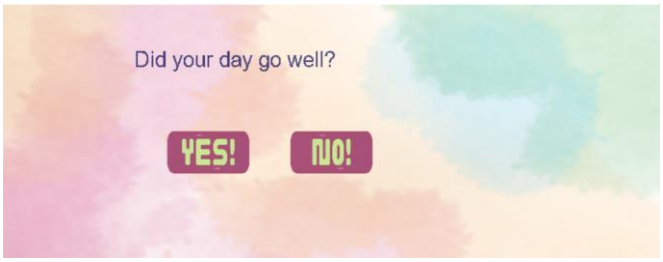
5. PROJECT DESCRIPTION:

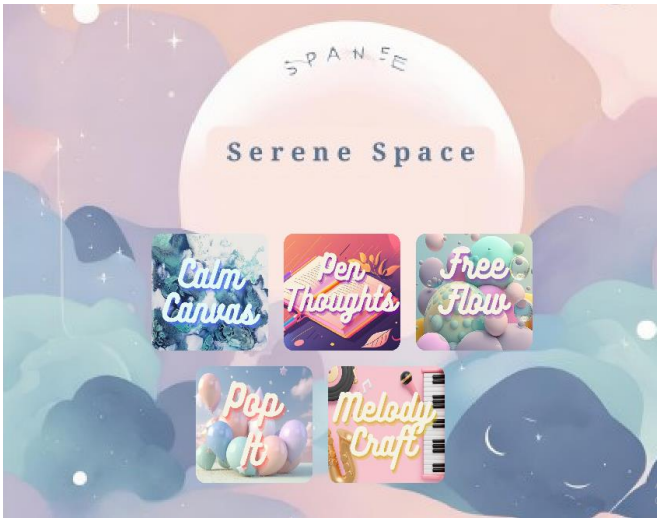
We have developed a captivating stress-relief application for our project. Upon launching the program, users are guided through an interactive decision-tree questionnaire using binary trees. Based on their responses, the system tailors recommendations, allowing users to decide on the stress-relief activity that best suits their preferences. The questionnaire leads them through a series of yes/no questions, eventually presenting the features of our application:

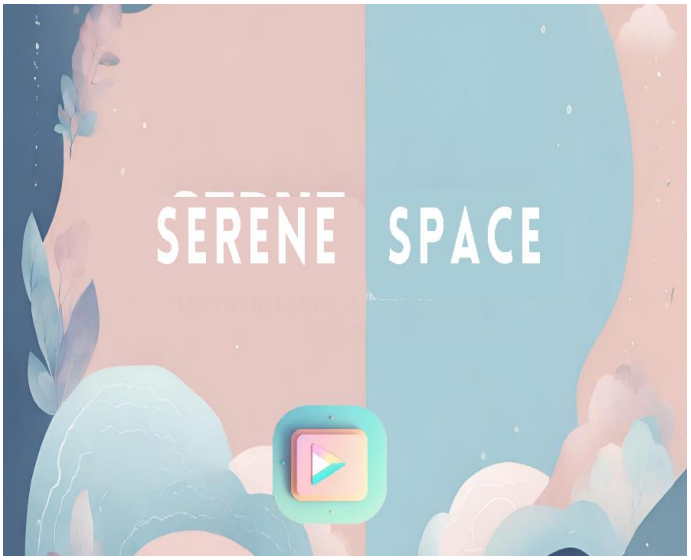
1. **Paint:**
    - Unleash your creativity with a digital canvas and various drawing tools.
2. **Flow Free:**
    - Engage your mind with a soothing puzzle game connecting colored dots.
3. **Journaling:**
    - Explore the therapeutic benefits of digital journaling in a calming environment.
4. **Balloon Pop:**
    - Release stress by popping virtual balloons in a fun and interactive game.
5. **Piano:**
    - Create beautiful melodies and unwind with a virtual piano experience.

6. HOW TO USE:

NAVIGATING THROUGH THE MAINSCREEN:







1- Begin your stress-relieving journey by pressing the "Play" or "Proceed" button displayed on the initial screen.

2- A questionnaire will then prompt a series of questions; to which you can respond by clicking either the "Yes" or "No" buttons.

3- After completing the questionnaire, the main screen will unveil a selection of five engaging activities: "Balloon Pop," "Paint," "Flow Free," "Journaling," and "Piano."

4- Choose the game that suits your mood or preferences by selecting the respective option.

INSTRUCTIONS FOR CALM CANVAS:

Pencil Tool: Utilize the pencil tool to draw freehand lines and sketches with precision using different sizes.

Eraser Tool: Correct any mistakes or unwanted elements

Colors: Click on the color options available to select the hue that suits your creative vision.

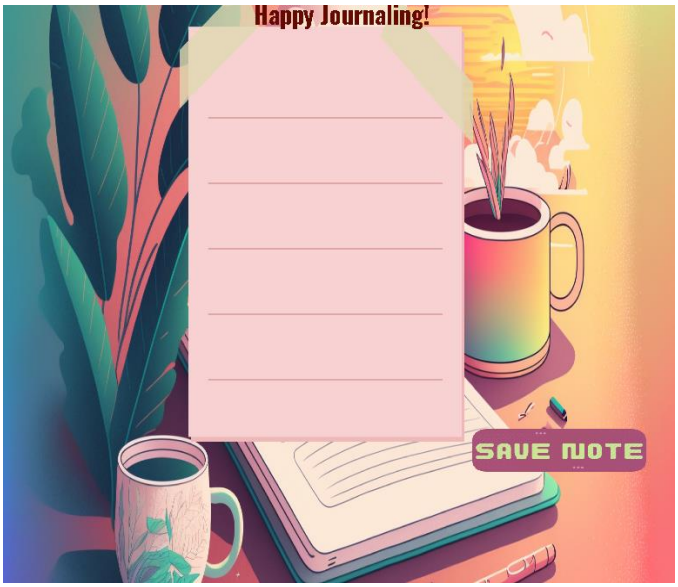
Brush: Experiment with brush to add texture and depth to your artwork and play with different sizes.

Bucket Tool: Select a color, click on an enclosed region, and fill the space.

Undo and Redo: Easily backtrack or reapply changes with a simple click.

Basic Shapes: Access a variety of basic shapes—such as circles, squares, and triangles—t to construct designs.

INSTRUCTIONS FOR PEN THOUGHTS:



You can:

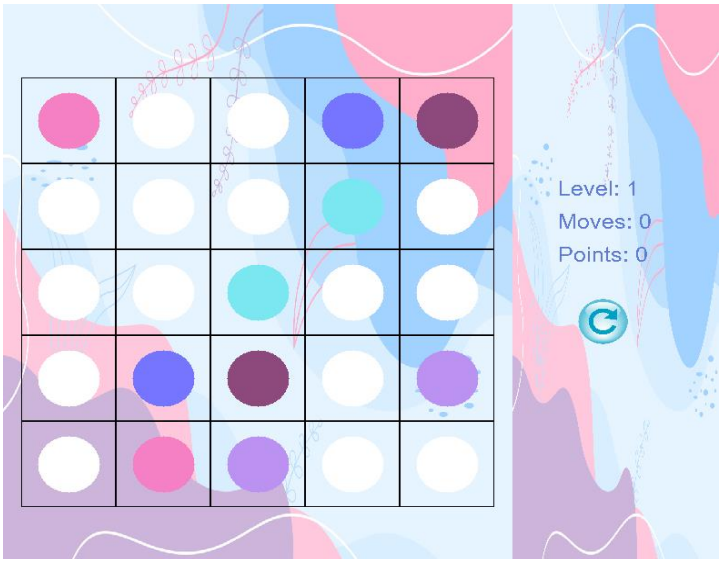
1- Create notes
2- View notes
3- Delete notes

1- Click on "Create Node" to start composing a new journal entry.
2- Input your thoughts and content in the provided text area.
3- Save your entry by clicking "Save Node."

Click "View Note" and put in an entry number to open an entry from a list of existing entries shown by their date and time.

4- Click "Delete Note" to remove entries.
5- Enter the entry number for deletion.

INSTRUCTIONS FOR FLOW FREE:



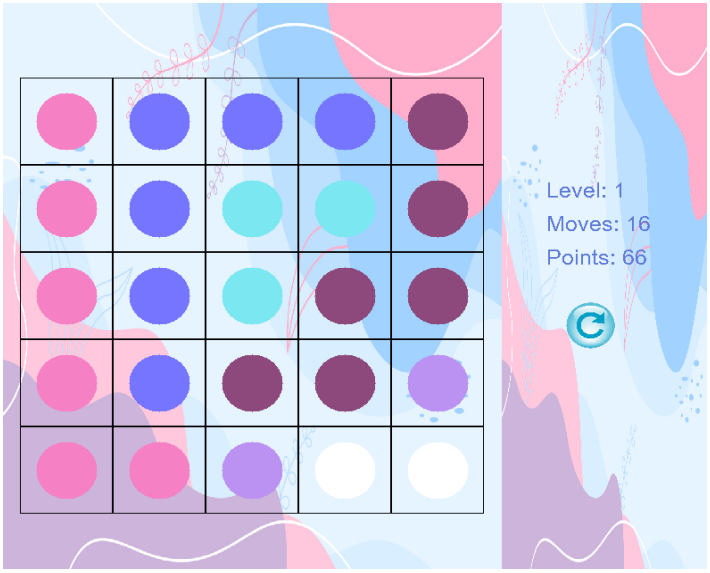
1- The goal of Flow Free is to connect matching colors with pipes to create a flow.

2- Connect the dots of the same color by drawing pipes between them.

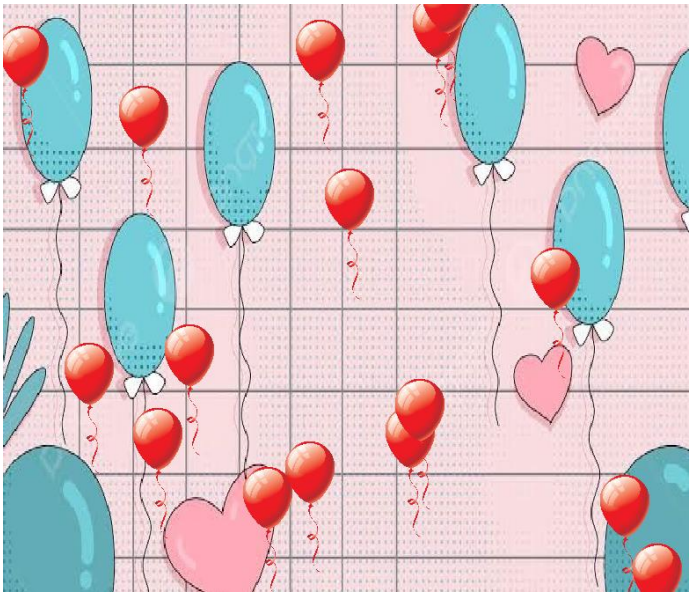
3- Pipes cannot cross or overlap, and each dot should be connected to its pair.

4- The number of moves you make is displayed, along with the points earned.

5- The game offers five levels, each with increasing complexity.



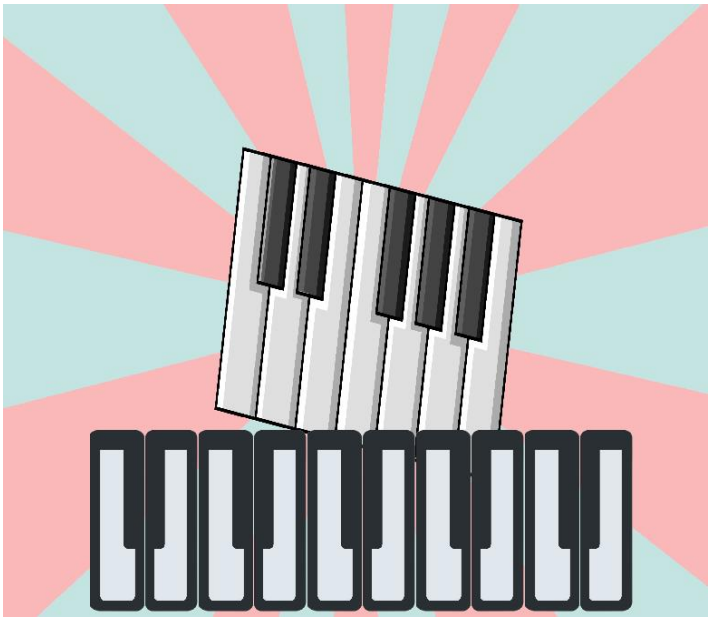
INSTRUCTIONS FOR POP IT:



1- Tap or click on the balloons to pop them as they float upwards on the screen.

2- As you pop balloons, positive affirmations such as "Good Job!" and "Wow!" will be displayed to encourage and uplift you.

INSTRUCTIONS FOR MELODY CRAFT:



1- Press the number keys from 0 to 9 on your keyboard to play different notes on the virtual piano.

2- Experiment with the keys to create your own tunes and melodies. Each number represents a unique musical note.

7. STEPS OF OUR SOLUTION:

Since the motive was to design a user-friendly and visually captivating stress relieving application, we decided to use the SDL Library for including graphics in the games. Along with SDL, we also used SDL_image for handling images in our program, SDL_ttf for dealing with fonts, and SDL_mixer, for incorporating audio in our games.

The application has six components including two games, a questionnaire, a paint app, a piano, and a note-taking diary that was divided among the members and created separately. Afterward, we implemented all the **data structures** including stack, queue, binary trees, circular and doubly linked list, combined all the data structures in a single DSA.hpp file, and included it in the abstract class “Stress Reliever” to use them in all the components.

In these components, certain constants are set, such as the window’s width and height, the dimensions of objects being used, and sounds incorporated.

The code adopts a sequential structure, initiating with the questionnaire module. Following its completion, users are directed to the main screen, displaying stress-relieving game options. Each game's source code is **encapsulated** in header files and seamlessly integrated into the "Menu" source code.

DETAILS OF THE GAMES:

1. FLOW FREE:

The C++ code implements the **"Flow Free"** game, emphasizing the effective use of data structures, including a **two-dimensional array** representing the game grid. The algorithmic backbone of the game involves a **backtracking** approach to efficiently navigate the grid, ensuring the proper connection of colored dots through drawn lines. This technique dynamically adapts to the increasing complexity of puzzles as players advance through levels. The data structures facilitate core algorithms that assess completed connections, validate moves, and determine level completion. Additionally, the scoring mechanism and considering moves, rely on variables extracted from the meticulously managed game data structures. While SDL handles graphics and input, the code underscores the significance of adeptly implemented data structures and the strategic application of backtracking algorithms for a fluid and engaging "Flow Free" gaming experience.

2. CALM CANVAS:

The “paint app” code presents a C++ implementation of a **paint application** using the SDL library, building upon the StressReliever base class. Central to this application is the PaintApp class, which employs various data structures to support diverse drawing tools and functionalities. The canvas is represented as a 2D vector of SDL_Colors, where each pixel's color is manipulated during drawing operations. The undo and redo functionalities are facilitated through a **Stack** data structure, enabling the reversal and restoration of drawing actions. User interactions, including mouse events and color selection, are efficiently managed through SDL data structures. In the Paint App, a **Queue** data structure is employed. This Queue serves a crucial role in efficiently managing and processing adjacent pixels when applying the bucket tool to fill an area with the selected color. The implementation reflects a thoughtful integration of data structures to optimize drawing operations and enhance the overall user experience within the paint application.

3. POP IT:

The “balloon pop” game introduces a **balloon-popping game** developed using the SDL library. The BalloonGame class, derived from the StressReliever base class, orchestrates the game mechanics. The initiation process involves setting up essential SDL components, such as sound, fonts, and background textures. Balloons are managed using a **linked list** structure, with each balloon represented by the Balloon structure. The game loop, housed in the run function, continually processes events, updates balloon positions, responds to user mouse clicks for popping balloons, and utilizes SDL functions for rendering the game window. Balloons exhibit random spawning and upward movement, triggering popping sounds and occasional affirmation messages upon interaction. Resource cleanup occurs systematically in the destructor and cleanup function**.**

4. MELODY CRAFT:

The provided C++ code introduces **a piano simulation** using the SDL library for graphics and sound. Notably, a linked list structure is leveraged to manage the functionalities of the

piano. The **LinkedList** template efficiently handles the initial X and Y positions of these keys, allowing dynamic organization and manipulation. Within the simulation loop, the program listens for key presses, plays corresponding notes, and updates the display. Visual feedback for piano keys involves loading button images and adjusting positions based on user input. There is a **Circular linked list** that manages SDL textures representing images. This circular linked list enables the smooth progression through a set of images, creating a continuous background effect reminiscent of video playback. This streamlined use of linked lists enhances the program's adaptability in dynamically managing piano keys during user interactions.

5. PEN THOUGHTS:

The "**Journaling Stress Reliever**" code presents an immersive stress-relief journaling experience implemented in C++ using the SDL library for graphics and input handling. The game incorporates key data structures, notably a **Doubly Linked List**, which efficiently manages journal entries and sorts them based on **their Date & Time**, facilitating seamless addition, viewing, and deletion of notes. The code utilizes dynamic arrays of SDL_Textures to enhance visual elements, with the LTexture class providing a structured approach for rendering text on the screen. Inheritance is employed, with the Journaling class inheriting from a base class (StressReliever), utilizing various SDL functionalities for event handling, text input, and overall user interaction. This combination of effective data structures and SDL features results in a compelling stress-relief journaling game that offers both functionality and visual appeal.

8. THE ADOPTED APPROACH:

We developed a diverse set of activities including Piano Game, Journaling, Flow Free, Paint, and Balloon Pop, our consistent approach revolved around leveraging the powerful capabilities of the SDL (Simple DirectMedia Layer) libraries. SDL served as a robust foundation for implementing these fun stress relieving activities and games, offering a

comprehensive suite of features that greatly simplified the application development process.

SDL's versatility played a pivotal role in creating an engaging and interactive user interface. The library seamlessly handled tasks such as rendering images and sprites, playing background music, integrating sound effects, and capturing user input. Whether it was the dynamic play of the Piano Game, the creative canvas of Paint, the strategic challenges of Flow Free, the immersive journaling experience, or the exhilarating balloon-popping action, SDL's functionalities allowed us to bring these diverse concepts to life with efficiency.

9. FURTHER EXPANSION OF THE PROJECT:

To elevate our "stress reliever" project, consider enhancing Paint with advanced tools like layers and varied textures. Introduce diverse balloon types in Balloon Pop for unique challenges. In Flow Free, integrate an intelligent hint system. Improve journaling through incorporating file handling. Improve the Piano Game with an option to record your own piano tunes expanded song options.