CSC 1300/ PROGRAM 3

HALLOWEEN PARTY PLANNER FALL 2024

DUE DATE:

Wednesday, November 6, 2024, by 11:59pm

- **You may turn in your submission up to two days late with a penalty of 10 points per day late.**
- After the two days have passed, the submission folder will close in iLearn, and you will not be able to submit.
- If iLearn marks your assignment as late, then the points will be deducted.
- **No programs sent outside of the iLearn assignment will be graded!**

HOW TO SUBMIT:

Upload a zip file named after your TTU username (**jdoe42_prog3.zip**) containing **halloween.cpp** and **Halloween.h** to the iLearn submission folder named **Program 3**.

PROGRAM OBJECTIVE:

Write a program that helps plan a Halloween party. The program will allow the planner to:

- Add, remove, modify invited guests
- Add, remove, modify party activities
- Print the available food and quantity
- Save to a Halloween party plan to a file

You must implement this program using branching, arrays (**NOT VECTORS**), and functions.

RULES TO REMEMBER:

- **Do NOT use ChatGPT, copilot, or any other generative AI** to produce code for this program. This is considered cheating, and you will be charged with academic misconduct and earn a ZERO for this programming assignment.
- **Do NOT work with a partner, friend, or classmate on this program!!** This will be considered cheating, and you will be charged with academic misconduct and earn a ZERO for this programming assignment. Get help from the Teaching Assistants/Mrs. Crockett/Mr. Vandergriff when you get stuck!!
- Include comments throughout your code and a comment block at the top of your source file containing the filename, author, title, and date
- Make your output neat, easy to read, and make sure everything is spelled correctly and uses proper grammar.
- Each programming statement should be on its own line, and you should use consistent indentions.
- **Do not use STL Vectors or STL Arrays in this program**. You are supposed to create arrays without having to include any libraries to do so.
- **Do NOT use programming constructs not yet taught in the class**. This means you can't use the following in your program:
    - Pointers
    - Structs
    - Objects/classes
    - Libraries not covered in class (can't use algorithm, map, etc.)

WHAT YOU WILL LEARN IN THIS ASSIGNMENT:

- Understanding of file input & output
- Understanding of while and/or do-while loops
- Menu systems in programs
- Using functions to organize your code
- Good programming practices in program format and commenting

PROGRAM SPECIFICATIONS:

HALLOWEEN.H

Your **Halloween.h** file should contain include guards, any C++ library #includes, using namespace std, and all programmer-defined function definitions created for this program.

HALLOWEEN.CPP

You will need 4 arrays for this program: **Guests, Activities, Food,** and **FoodQuantity**.

- You will also need a way to track how many elements are currently in your Guests and Activities array. I started my arrays partially filled.
- This program assumes that the food arrays are full.

PROGRAM MAIN LOOP

Your program should print out a menu that says:

```
~Time to plan my awesome Halloween party!~
1. Manage Guest List
2. Manage Activities
3. Check Food
4. Save Party to File
5. Exit Party Planning
```

USER SELECTS CHOICE 1

If the user selects to manage the guest list, you will need to call **manageGuestList(),** and show the currently invited guests.

Then you display another menu that says:

```
  Guest list:
        1. Dracula
        2. Steve Jobs
        3. Ms. Loch Ness
        4. Jeff
        5. Dr. Gannod
 1. Add Guest
 2. Remove Guest
 3. Modify Guest
 4. Exit Guest List
```

If the user chooses to **add a guest**, you will need to add a guest to your array.
If the user chooses to **remove a guest**, you will need to remove a guest from your array.
If the user chooses to **modify a guest**, you will need to adjust an existing guest in your array.

You will need to check the edge cases – what if our array is full and we try to add? What if it is empty and we try to remove?

If this happens, return a **-1** from the function, and print that an error happened in your main function.

```
   1
Enter the guest's name.
      spooky ghost man
Guest list:
        1. Dracula
        2. Steve Jobs
        3. Ms. Loch Ness
        4. Jeff
        5. Dr. Gannod
        6. spooky ghost man
 1. Add Guest
```

```
 2. Remove Guest
 3. Modify Guest
 4. Exit Guest List
      2
 Enter the guest you want to remove.
      4
Guest list:
        1. Dracula
        2. Steve Jobs
        3. Ms. Loch Ness
        4. Dr. Gannod
        5. spooky ghost man
 1. Add Guest
 2. Remove Guest
 3. Modify Guest
 4. Exit Guest List
```

USER SELECTS CHOICE 2

Choice two has the exact same flow as choice 1, but with a different array.
It will be called with the **planHauntedActivites() function.**
Then you display another menu that says:
You will need to check the edge cases – what if our array is full and we try to add? What if it is empty and we try to remove?

If this happens, return a **-1** from the function, and print that an error happened in your main function.

```
1. Manage Guest List
2. Manage Activities
3. Check Food
4. Save Party to File
5. Exit Party Planning
2
Activities:
        1. Apple Bobbing
        2. Costume Contest
        3. Pumpkin Carving
        4. Ghost Stories
        5. Pin the tail on the ghoul
 1. Add Activity
 2. Remove Activity
 3. Modify Activity
 4. Exit Activity List
1
Enter the activity to add:
crying
Activities:
        1. Apple Bobbing
        2. Costume Contest
        3. Pumpkin Carving
        4. Ghost Stories
        5. Pin the tail on the ghoul
        6. crying
 1. Add Activity
 2. Remove Activity
 3. Modify Activity
 4. Exit Activity List
```

USER SELECTS CHOICE 3

If the user selects display food, we want to print the current options available, and the number of servings it has.

```
1. Manage Guest List
2. Manage Activities
3. Check Food
4. Save Party to File
5. Exit Party Planning
3

Witch Finger Cookies has 7 servings prepared.
Zombie Meatloaf has 6 servings prepared.
Eyeball Punch has 12 servings prepared.
Spider Web Cupcakes has 15 servings prepared.
1. Manage Guest List
2. Manage Activities
3. Check Food
4. Save Party to File
5. Exit Party Planning
```

USER SELECTS CHOICE 4

If the user selects Choice 4, loop over all the arrays and output them to a file.
The output should be formatted as the following:

fileExample.txt

```
Guest List:
      Dracula
      Steve Jobs
      Ms. Loch Ness
      Dr. Gannod
      spooky ghost man
Planned Activities:
      Apple Bobbing
      Costume Contest
      Pumpkin Carving
      Ghost Stories
      Pin the tail on the ghoul
Food:
      Witch Finger Cookies: 7
      Zombie Meatloaf: 6
      Eyeball Punch: 12
      Spider Web Cupcakes: 15
```

FUNCTIONS

`int manageGuestList(string[], int);`

- Takes a string array and an integer as an argument.
- Enters a do-while loop
- Returns a 0 if operation was successful, returns a -1 if something failed.

`int planHauntedActivities(string[], int);`

- Takes a string array and an integer as an argument.
- Enters a do-while loop

- Returns a 0 if operation was successful, returns a -1 if something failed.

```
void trackFood(string[], int[]);
```
- Takes a string array and an integer array as an argument.
- Should print all the food and serving amounts of food the the screen

```
void printMenu();
```
- Comfort function – optional. Prints the main menu

void saveToFile(string[], **int**, string[], **int**, string[], **int**[]);
- Loop over all of the arrays and print them to an output file. You can hardcode this file, or take it as user input.

OPTIONAL CHALLENGES / THINGS TO THINK ABOUT

The **`manageGuestList`** and **`planHauntedActivities`** functions work the same, except with different output.

Can you write a generic function to work with both?