Projects for Interapt Training 2.0

Content:

Dungeons & Dragons (D&D) Character Creator:

Create a D&D character creator with the following parameters:

1) Contains a model for Strength, Dexterity, Intelligence, Wisdom, Charisma, & Constitution.
2) User should see the printed result of three separate 6-sided dice rolls that will determine the character's Strength, Dexterity, Intelligence, Wisdom, Charisma, & Constitution.
    a) Should use a single method that operates as a single 6-sided dice roll three times, not three different methods that do the same thing.
3) Each value should be stored in an array.

Content Covered:

- Introduction to Models.
- Introduction to Object Oriented Programming.
- Introduction to Random Number Generation.
- Introduction to Pushing to an Array.

Notes:

I could make this a Yahtzee game? might be better but it'd be kind of boring. Just if D&D is too weird.

Concert Lister:

Create a program with the following parameters:

1) Ask the User to input the concerts they have been to.
    a) When a User writes “done” it’ll stop taking input.
2) Takes the User’s input and prints to an array.
3) Prints the contents of the array.
4) Ask the User to rate the concerts they’ve been to
    a) Will print out the concert name and ask the User to rate it on a scale of 1 to 5.
5) Print out the concerts by rank.

Content Covered:

- Introduction to User Input.
- Introduction to arrays.
- Introduction to pushing to an array.
- Introduction to loops.

Notes:

Name Fun:

Guide should include details on how to create a program that:

1. Ask a User to print their name
2. Takes that input.
3. Prints the User's name backwards using loops.
4. Prints the sum of letters in the User's name.
    a. ex: the name "Eagle" would print out 5.
5. Prints the alphabetically arranged letters of the User's name.
    a. ex: The name "Eagle" would print out "A, E, E, G, L."

Content Covered:

- Introduction to user input.
- Introduction to loops.
- Introduction to arrays.
- Introduction to pushing to arrays.

Notes:

Attendance Roll Call:

Create a program with the following features:

1) A menu for User to add a name, see all added names, & look up names.
2) Added names will be arranged alphabetically using bubble sort.
3) If User looks up names, printing out a name will find all instances of that name.
    a) If multiple individuals are named “Kevin”, should print out all instances.
    b) User can find a name with either first or last name.
4) User can delete a name.

Content Covered:

- Loops in loops.
- User input.
- Adding to an array
- Deleting an item from an array
- Searching through a data set

Notes:

Rock, Paper, Scissors Project:

Create a walkthrough for a Rock, Paper, Scissor Project with the following parameters:

1) User able to input their choice of Rock, Paper, or Scissor.
2) The program responds with a randomized choice
3) The program prints both responses.
    a) ex: “User picked: SCISSORS” / “Computer picked: ROCK”
4) User sees a printed message, if they won, lost, or drew.

Content Covered:

- User input
- Using a random number generation.
- “Logical and” Operator
- “Logical not” Operator

Rock, Paper, Scissors Project Part 2:

With your existing Rock, Paper, Scissor Project add the following parameters:

1) Add a Menu for the User to see their statistics, which contains
    a) Count of how many times the User has played the game.
    b) User win percentage, loses, wins, & draws.
2) The program should count the amount of times a User has chosen Rock, Paper, Scissors & make its choice directly based on the opposite of what the User has chosen.

Content Covered:

- User input
- Using a random number generation.
- “Logical and” Operator
- “Logical not” Operator
- Pushing to arrays
- Data collection/comparison

Notes:

Converter Project:

Create a walkthrough that contains the following:

1) Create an inches to feet converter where User inputs a number of inches and it converts that number to feet.
    a) ex: If User inputs “75” the program’s response should be: “6ft 3in”.
2) Create a seconds to minutes converter where User inputs a number of seconds and it converts that number to minutes.
    a) ex: If User inputs “75” the program’s response should be “1min 15seconds”.
3) Create a universal converter that ask the user what they’d what to convert “inches” or “seconds” and can take either input and convert to feet or minutes. It can also be added too.
    a) ex: If a client wants to add a feet to yards method, it should be easy to add to the method.

Content Covered:

- Object Oriented Programming
- Reusable code
- Basic math

Notes:

Hex-code guessing game:

Create a guessing game app with that implements the following:

1) Randomly generate a hex sign for Red, Green, & Blue.
    a) Use enums to store those (A, B, C, D, E, F) values.
    b) Print the color of the randomly generated hex sign.
2) User can input values for Red, Green, & Blue.
3) The program will compare the User's guess for each value and to the randomly generated one.
    a) For each value, the program will tell the User if it's too high or too low or the correct value.
    b) The color the User guessed will be printed.
    c) The User will only have five attempts.
        i) After the fifth attempt the program will reveal the hex-code.
        ii) User can choose to restart
            (1) Restart gives a new randomly generated hex-code.

Content Covered:

- Hex codes
- User input
- Data comparison

Notes:

Obviously this game is impossible to win but eventually it'll teach an eye for color. So that someone can see the

State Capitol Testing Game:

Create a hash-map/dictionary for all fifty states in America that contains the name of the state and that state's capital.

Using that hash-map/dictionary create a state capital testing game that implements the following:

1) The program will ask the User to input the state capital of a state
2) Take a value from a User (a User's answer).
    a) If the User is incorrect, the appropriate Capital will be printed.
    b) If the User is correct, they will receive a congratulatory message.
    c) User's right & wrong answers should be kept in an array.
3) The User should never be asked about the same State twice.
4) When the User has been asked about all the 50 American States, the program will print out the User's grade.
    a) A => 90%
    b) B => 80% && B =< 89%
    c) C => 70% && C =< 79%
    d) D => 60% && D =< 69%
    e) F =< 59%
        i) This content should be stored as an enum.
5) User should see a print out of every state they didn't know the capital for.

Content Covered:

- Hashmaps/Dictionaries
- Enums
- Loops
- Pushing to an array
- User input

Notes:

Track Game:

Create a track & field game, with the following features:

1) Contains a model for a runner with the following:
    a) Personal Best Time (6mins), Personal Worst Time(8mins), Personal Best Jump (7 feet), Personal Worst Jump (3 feet).
2) User should be greeted with a Menu with the following options:
    a) Practice Jumping
    b) Practice Running
    c) See Previous Times and Jumps
3) If User chooses Practice Running they will be given a time between the Runner's personal best time & personal worst time.
    a) Use your converter from the previous project to print out the Runner's time.
        i) Watch out for 7 mins, 98 secs!
    b) Each time should be added to an array of previous times.
    c) If the time returned is smaller than any in the array the User should know they have a new "Best Time."
4) If User chooses Practice Jumping they will be given a jump between the Runner's personal best jump & personal worst jump.
    a) Use your converter from the previous project to print out the distance of the Runner's jump.
        i) Watch out for 5 feet, 98 inches!
    b) Each jump should be added to an array of previous times.
    c) If the time returned is smaller than any in the array the User should know they have a new "Best Jump."
5) If User chooses See Previous Times and Jumps, they should see:
    a) All the User's run time and jump distances.
    b) A column that shows their average Jump distance, average mile times.
    c) The personal best time and jump.

Content Covered:

- Models
- Loops
- Data collection/comparison
- Averages

Notes: