|  |  |
| --- | --- |
| **Final Project Planning Guide** | Name: |
| **Remix Step 1: Review your programs from the mission pack** | |
| What programs / missions were your favorite? What did you like about them? |  |
| What programming concepts do you feel you understand the most? |  |
| What programming concepts do you need help with? |  |
| **Remix Step 2: Remix Project Concept** | |
| Look over the remix suggestions from your favorite projects. Discuss with your team. Decide on a final project that will both interest and challenge you. Describe what your final project will do: |  |
| **Remix Step 3: Plan your code. What variables will you use in the project?**  Fill out the charts below. Use another piece of paper to design your program with an algorithm. | |
| What variables and lists will you use in the project? Fill in the chart. You do not need to fill in every line, or you can add more. | |  |  | | --- | --- | | Variable / List Name | What it will be used for: | |  |  | |  |  | |  |  | |  |  | |
| What dictionaries or lists will you use? How will they be created? What information will they contain? |  |
| What will you use for input (input() or buttons pressed). Describe the input and expected outcome. |  |
| What functions will you write? Describe each one.  Add more rows as needed. | |  |  | | --- | --- | | Function name | What it will do | |  |  | |  |  | |  |  | |  |  | |  |  | |
| Extras:  What else will you need for your program?  (sound, LEDs, etc.) Describe extra code you will use in the space provided: |  |
| **Remix Step 4: Write your code** | |
| Start a new file. Use the sandbox  when you write the code. Write just a few lines at a time and test often. You can choose which 3D environment you want for the remix project. | |
| **Remix Step 5: Commenting and feedback**  Documentation: Make sure your code is readable by adding blank lines and comments to explain your code. | |
| **Peer feedback:** Get feedback from two (or more) people. You can be one of the peer reviewers. | |
| Peer Review #1 Name: |  |
| Go through the checklist. Are all requirements met? If not, list any missing criteria. |  |
| What do you like about the program – be specific! |  |
| Give at least one suggestion. Begin with “what if” or “maybe you could” |  |
| Peer Review #2 Name: |  |
| Go through the checklist. Are all requirements met? If not, list any missing criteria |  |
| What do you like about the program – be specific! |  |
| Give at least one suggestion. Begin with “what if” or “maybe you could” |  |
| Review the comments. Then take time to improve or add to your project. | |
| **Post-Mission Reflection** | |
| What did you change in your project after reading the feedback? |  |
| What do you like most about programming? |  |
| What do you find the most challenging about programming? |  |
| How have your attitudes or feelings about computer science changed during this mission pack? |  |

|  |  |  |  |
| --- | --- | --- | --- |
| **Final Project Rubric** | | | |
| **Requirement** | **No evidence ←—-------------------------------------------------------→ Mastery** | | |
| **Programming Conventions** are followed | * Variable names aren’t descriptive * Function names aren’t descriptive * Code blocks inconsistently indented * Capital letters used * Code is not organized into sections |  | * Variable names are descriptive * Function names are descriptive * Code blocks consistently indented * Use of small letters (not capital) * Code is organized into sections |
| **Documentation and Readability** | * No comments are used. * Code is difficult to read because no blank lines were used, or too many blank lines were included. |  | * Frequent and descriptive comments are used regularly. * Blank lines are used to help with readability. |
| **Use of Variables and constants** | * “Magic Numbers” or literal values are used in the code. * Data isn’t tracked or updated (no counters, states, conversions, etc.). |  | * Constants are used (no ‘magic numbers’) * Variables are used for storing, keeping track of and updating data. * Global and local variables are used. |
| **Use of Functions** | * No plan or algorithm to follow. * Everything in one main program. * Long sections of code. * Functions use all global or all local variables. * Functions don’t take parameters. |  | * Code is divided into smaller sections that accomplish a task. * Parameters are used as needed. * Local and global variables are used. * Functions return a value as needed. |
| **Use of Inputs**  Buttons and sensors | * Neither button is used for input. * No sensors are read or used. |  | * At least one button is used. * At least one sensor is used to give input. * Conversion of raw data is performed. |
| **Algorithms and Programming** | * No algorithms identified or used. * Program performs the same for every execution, without input. * Lists and tuples are not utilized when they would simplify the code. * Debugging practices are not used and code contains errors. |  | * Algorithms are used to manipulate data and get results. * Data is used to inform decisions. * Lists and tuples are used to simplify data collection and implementation. * Debugging practices are used to correct errors in code and logic. |
| **Control Structures** | * Program does not have any if or if/else or if/elif/else statements. * Program does not use any loops. |  | * Loops and if statements are used to control the flow of execution. * Conditional and logical operators are used appropriately. |
| **Use of Outputs**  LEDs, speaker, motors | * No output is produced. |  | * One or more outputs are used to convey data or perform a task. |
| **Collaboration** | * Students work independently or uncooperatively on a team. |  | * Students work collaboratively with shared tasks in their team to complete the project. |
| **Synthesis / Purpose** | * No clear purpose for the program. * Program does not incorporate learning across the mission pack. |  | * Purpose of the program is clearly stated. * Program combines learning, concepts and code from several missions. |
| **Code Completion** | * Code will not run or doesn’t complete the task correctly. |  | * Code runs and accomplishes its task without any errors, including logic. |