Project 1 Action Plan

In each project you code, you must think about how you want the project to look and what you want it to do. Then, write an *action plan* — a sequence of steps — to complete your project.

In this book, a suggested action plan is presented for each project, but you are encouraged to use your own creativity in adjusting the plan to build the project the way you want. Remember, *you* are the programmer!

Brainstorm the Art Toy

- [ ] Brainstorm the goal and design of the toy.
- [ ] Start a new project.

Lay Out and Code the Toy Page

- [ ] Add a title for the toy.
- [ ] Create a turtle object, which will do the drawing.
- [ ] Test various primitives in the Command Center.
- [ ] Create buttons to execute primitives; add new buttons.
- [ ] Write procedures in the Procedures pane, and then add buttons to execute procedures; add new procedures and buttons.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Creating a turtle object, primitives versus procedures, nested procedures, moving an object (forward, backward, right turn, left turn), adding color to an object, adding color to the background, cleaning the background, going to home position, making buttons, repeating instructions, creating curves, coordinates, degree angles, generating random numbers, creating a graphical user interface.

Project 2 Action Plan

Mini Golf is a fun game in the real world and in the video game world, too! Creating this game requires painting a green, hole, and water trap; coding controls for aiming and hitting the ball; and coding reactions to the hit (falling in the water, bumping off an obstacle, and sinking the ball). Here's an action plan.

Brainstorm the Game

- [ ] Brainstorm the goal and design of the game.
- [ ] Start a new project.

Lay Out and Code the Game Page

- [ ] Paint the green, water trap, and hole.
- [ ] Add a title.
- [ ] Create two objects, a golf ball and an obstacle.
- [ ] Set the starting position of the golf ball.
- [ ] Create controls for aiming and hitting the ball.
- [ ] Code the ball to bump off an obstacle.
- [ ] Write universal color conditionals for the water trap and hole.
- [ ] Write a `watertrap` procedure, which executes when the ball touches the water.
- [ ] Write a `win` procedure, which executes when ball touches the hole.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Painting a background, modifying a shape, adding a shape to an object, setting starting coordinates, setting absolute versus relative headings for objects, adding wait time, smooth motion via gliding, rebound motion following a collision, universal color conditional, hide object, show object, making announcements.

Project 3 Action Plan

Sketcher Etcher is a simple drawing toy that you code using paint tools, some drawing commands, and a special set of commands that let you define the boundaries of the drawing area.

Brainstorm

- [ ] Brainstorm the design of the game.
- [ ] Create a new project.

Lay Out and Code the Toy Page

- [ ] Draw and freeze the background.
- [ ] Create one turtle, put its pen down (`pd`), and add a shape to it so that it looks like a drawing stylus.
- [ ] Add buttons to make the stylus draw left, right, down, and up.
- [ ] Write boundary conditions to prevent the stylus from drawing on the left and right sides of the frame.
- [ ] Write boundary conditions to prevent the stylus from drawing on the bottom and top sides of the frame.
- [ ] Create a button to clean the drawing off of the Sketcher Etcher.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Freezing the background, setting boundary conditions using inequalities, using inverse motion.

Project 4 Action Plan

Horse Race is a fun simulation featuring racing horses on a racetrack. Follow this action plan to get racing!

Brainstorm

- [ ] Brainstorm the design of the game.
- [ ] Create a new project.

Lay Out and Code the Simulation Page

- [ ] Paint the simulation page, which is where the horse race takes place.
- [ ] Create one horse character and then add primitive commands to the OnClick field to create its racing motion.
- [ ] Make copies of the horse to add a total of four horses to the race.
- [ ] Create a Start button to align the horses at the start of the race.
- [ ] Set up horses to recognize the red finish line by using OnColor; add a `stopall` command to stop the simulation.
- [ ] Create a button that activates ClickOn for all the horses at the same time and starts the race.
- [ ] Create audio of trotting horse hooves to play during the race.
- [ ] Make a Trot button to continue playing the trotting melody until a horse crosses the finish line.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Using animated shapes, aligning objects, random speed, activating an object via a mouse click, activating all objects at the same time, creating and playing digital audio, parallel execution of commands, object specific color conditional, stopping program execution.

Project 5 Action Plan

Winter Wonderland is a fun project in which you create an active snow scene. Following is a simple action plan to complete the project:

Brainstorm

- [ ] Brainstorm the design of the game.
- [ ] Create a new project.

Lay Out the Scene Page

- [ ] Paint the scene page by adding a tree and snow on the ground.
- [ ] Create one bulb character, add the `blink` procedure to it, and then make multiple copies of the bulb and arrange them on the tree.
- [ ] Create one snowflake character, add the `fall` procedure to it, and then make copies of the snowflake and arrange them in the sky.
- [ ] Import winter-themed music from the web.

Code the Scene Actions

- [ ] Write a `blink` procedure to change the colors of the bulbs on the tree.
- [ ] Write a `fall` procedure that makes the snowflakes gently fall.
- [ ] Create a Winter button, which clicks on all the light bulbs and snowflakes at once.
- [ ] Write a `night_and_day` procedure to slowly change the background color of the scene. Then create a Night and Day button to execute the procedure.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Creating new animated shapes, time-based execution of commands, irregular falling motion, importing and playing music, random shape selection.

Project 6 Action Plan

Sports Vote can be used to collect votes for any type of candidate. Create your machine as follows:

Brainstorm

- [ ] Brainstorm the design of the voting machine.
- [ ] Create a new project.

Lay Out and Code the Voting Machine Page

- [ ] Add a sports-themed background to the workspace and paint a semi-opaque rectangle over the background.
- [ ] Add text boxes and text for the title and directions explaining how to vote and how to interpret the results.
- [ ] Create three turtle candidates.
- [ ] Add a sports shape to each turtle candidate.
- [ ] Write a `startup` procedure that resets the voting machine when it's opened.
- [ ] Make a bell sound using the Create a Melody button.
- [ ] Write a `getvote` procedure and add it to the OnClick field of each turtle. When the user clicks a turtle to vote, the `getvote` procedure plays the bell sound and increases the size of that turtle.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Adding a MicroWorlds EX background, working with the size variable, setting initial variable value, changing variable value according to user actions, adding text directions to the graphical user interface, creating and playing digital audio.

Project 7 Action Plan

Constructing a simulation (sim) is a fun process that goes through several phases. First think about how you guess the real world works. Then design and code a simulation that represents that guess. Look at the data your simulation produces and see how that data compares with the real world. Make revisions to your model until you have simulated reality as much as possible. Then play with your simulation to see how it behaves under a variety of conditions. Here's an action plan for building the maternity ward:

Brainstorm the Simulation

- ☐ Brainstorm the goal and design of the simulation.
- ☐ Start a new project.

Lay Out and Code the Sim Page

- ☐ Apply a hospital-themed background.
- ☐ Add a title.
- ☐ Create male and female characters.
- ☐ Make variables for baby gender, total number of girls, and total number of boys.
- ☐ Write an `initialize` procedure and make an associated button.
- ☐ Write a `reproduce` procedure and make an associated button.
- ☐ Write a `make_10_babies` procedure and make an associated button.

Debug and Enhance

- ☐ Save, test, and debug.
- ☐ Enhance.

Key Ideas

Simulation building, initializing a model, creating new variables, random selection of characters, `IF-THEN` conditionals, discrete variables, counters, nesting procedures, exploring frequency distribution, repeating instructions.

Project 8 Action Plan

Number Guessing Computer is a guess-and-check game that uses simple comparison logic to guide a player toward guessing a secret number.

Brainstorm

- [ ] Brainstorm the design of the game.
- [ ] Create a new project.

Lay Out and Code the Game Page

- [ ] Paint a background, including the computer.
- [ ] Create a title text box.
- [ ] Create a secret number variable (`secnum`) in the form of a text box.
- [ ] Make a slider to set the highest possible value of the secret number for a round of the game.
- [ ] Create a turtle and paint four shapes for it: blank, too low, too high, and correct.
- [ ] Write a `compchoosesecret` procedure for the computer to randomly choose a secret number and make an associated button to execute the procedure.
- [ ] Write a `checkguess` procedure to ask the player to input a guess, compare the guess to the secret number, and report the result to the player.
- [ ] Create a tune that plays when the player makes a correct guess.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Hide the secret number variable (text box).
- [ ] Enhance.

Key Ideas

Adding a slider, creating a random number given a slider-set maximum, accepting number input, the law of trichotomy, `IF-THEN` conditionals, changing character shape according to outcomes, nesting procedures, recursion, making announcements, creating and playing digital audio.

Project 9 Action Plan

Monster Mashup can take a variety of forms. Anything that involves mix-and-match options will work . . . but the underlying program is the same no matter what theme you choose! Here's a simple action plan to construct your toy:

Brainstorm

- [ ] Brainstorm the goal and design of the toy.
- [ ] Create a new project.

Lay Out and Code the Toy Page

- [ ] Color the background and add a title.
- [ ] Create and name four turtles for the four different monster parts.
- [ ] Create and name shapes for each hair, face, body, and feet part.
- [ ] For each monster part, add a drop-down list showing the shape choices.
- [ ] Code a selection procedure for each monster part that assigns a selected shape to the turtle, and make a button to execute the procedure.
- [ ] Code a `mashup` procedure to randomly select shapes for monster parts and put them all together, and make a button to execute the procedure.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Creating new shapes, making drop-down selectors and buttons, `IF-THEN` conditionals, changing character shape according to user selection, random shape selection, writing more complex procedures.

Project 10 Action Plan

Viral Outbreak can model anything that transmits "virally" from illness to information. Create your simulation as follows:

Brainstorm

- [ ] Brainstorm the design of the game.
- [ ] Create a new project.

Lay Out and Code the Simulation Page

- [ ] Paint a simple pattern and a hospital image on the background.
- [ ] Create a turtle.
- [ ] Write an `initialize` procedure and make an Initialize button.
- [ ] Make a Clone button and write an associated `clone` instruction.
- [ ] Make a 1 Sick button and write an associated instruction that will set the color of one turtle to red, indicating it's infected.
- [ ] Write an `infect` procedure and add it to the OnTouching field of the turtle.
- [ ] Write a health command and add it to the OnColor field of the turtle.
- [ ] Write a `roam` procedure and make a Roam button.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Simulation building, painting a background, initializing a model, adding color to an object, current object, cloning, random scattering of objects, attribute passing between specific objects, roaming motion, universal color conditional.

Project 11 Action Plan

UFO Pilot is a flying game that invites a player to navigate a side-scrolling asteroid field. Follow this action plan to construct the game.

Brainstorm

- [ ] Brainstorm the goal and design of the game.
- [ ] Create a new project.

Lay Out the Game Page

- [ ] Add a background to the workspace.
- [ ] Create a UFO character.
- [ ] Create asteroid cluster characters.
- [ ] Create a score variable.

Code the Gameplay

- [ ] Write a `reset` procedure to reset the game and make an associated button to execute the procedure.
- [ ] Code the pull of gravity on the UFO and add it to the OnClick field of the UFO's backpack.
- [ ] Code the crash between the UFO and asteroids and add it to the OnTouching field of the UFO's backpack.
- [ ] Write a keyboard-controlled `fly` procedure for the UFO.
- [ ] Write a `checkpass` procedure to score the motion of the asteroids passing the UFO.
- [ ] Code the motion and scoring of the upper asteroid clusters and match the motion of the lower asteroid clusters to their top partners.
- [ ] Create a Go! button to start the game.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Applying background images from the web, making a scoring variable, side-scrolling motion, using the keyboard to control a character in one direction, simplified gravity, mirrored motion, collision recognition.

Project 12 Action Plan

Squid Ink is a shooter style game that allows a squid (the shooter) to fire an object (an ink cloud) at a target (the shark). Follow this action plan to construct the game:

Brainstorm

- [ ] Brainstorm the goal and design of the game.
- [ ] Create a new project.

Lay Out the Game Page

- [ ] Add a background to the workspace.
- [ ] Create a squid character.
- [ ] Create an ink cloud character.
- [ ] Create a shark character.

Code the Gameplay

- [ ] Write a `go` procedure to start the game action, and make an associated button to execute the procedure.
- [ ] Write a `swimshoot` procedure and add it to the OnClick field of the squid's backpack.
- [ ] Write a `shoot` procedure.
- [ ] Record "Argh, Inked Again" and "Chomped" sounds.
- [ ] Write a `geteaten` procedure and add the procedure to the OnTouching field of the squid's backpack.
- [ ] Write a `hit` procedure and add it to the OnTouching field of the ink's backpack.
- [ ] Write a `float` procedure and add it to the OnClick field of the shark's backpack.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Random scattering of objects, using the keyboard to control a character in four directions, carrying cargo, nesting procedures, multidirectional shooting, recording and playing sounds, launching a parallel process, targeted movement towards an object, collisions and conditionals between specific objects.

Project 13 Action Plan

Rock, Paper, Scissors is a fairly simple game to create. Just follow this action plan:

Brainstorm

- [ ] Brainstorm the design of the game.
- [ ] Create a new project.

Lay Out the Game Page

- [ ] Paint the game page.
- [ ] Add text to the game page.
- [ ] Create characters (rock, paper, scissors).

Code the Gameplay

- [ ] Create a button for each character, and write code for the player to choose that character.
- [ ] Write a `compchoose` procedure for the computer to choose a character randomly.
- [ ] Write a `checkwin` procedure with compound conditionals to determine who wins the match.

Debug and Extend

- [ ] Save, test, and debug.

Key Ideas

Compound conditionals (`IF AND THEN`), nesting procedures, random selection of characters, commenting, text-to-speech production.

Project 14 Action Plan

Where's Wallace Walrus? can feature any characters and setting in which the player must search a scene to find the target character.

Brainstorm

- [ ] Brainstorm the design of the game.
- [ ] Create a new project.

Lay Out and Code the Search Page

- [ ] Name the search page.
- [ ] Paint a simple background on the search page.
- [ ] Create a main character turtle, name it `walrus`, and create an original shape for it.
- [ ] Create distracter shapes.
- [ ] Make a slider to create a variable number of distracters in the population.
- [ ] Write a `populate` procedure and then make a button to execute that procedure.
- [ ] Write a `getready` procedure to clear the search page and make a button to execute it.
- [ ] Create a text box for page directions.
- [ ] Write a `foundhim` procedure and add the procedure to the OnClick field of the main character.

Lay Out and Code the Splash Page

- [ ] Create and name the splash page.
- [ ] Paint the background and stamp characters.
- [ ] Create text boxes to explain the back-story of the search-and-find.
- [ ] Make a Let's Play button to advance to the search page.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Adding a splash page, cloning, creating a variable number of objects set via a slider, scattering of objects, removing objects, mouse-click selection of objects, `IF-THEN-ELSE` conditional, removing objects, mouse-click selection.

Project 15 Action Plan

Traffic Dodge is one of the more complex games in this book. To better understand the steps in making the game, it's a good idea to create an action plan.

Brainstorm

- [ ] Brainstorm the goal and design of the game.
- [ ] Create a new project.

Lay Out the Level 1 Game Page

- [ ] Paint and name the Level 1 game page.
- [ ] Create a frog character.
- [ ] Create traffic characters that serve as obstacles for the frog.
- [ ] Make traffic move on the page.
- [ ] Create a project variable to track lives remaining.
- [ ] Show the value of the lives remaining variable.

Lay Out the Splash Page

- [ ] Create the splash page.
- [ ] Write a `start` procedure and then make a Start button to execute the procedure.
- [ ] Add a page transition from the splash page to the Level 1 game page.

Code the Gameplay

- [ ] Write a `start` procedure to set up the game and move to the Level 1 game page and then make a Start button to execute the procedure.
- [ ] Write a `play` procedure to initiate game play.
- [ ] Write a `jump` procedure for keyboard control of the frog and then add this procedure to the frog.
- [ ] Write a `hit` procedure and then add this procedure to the frog.
- [ ] Write an `endgame` procedure that executes when the frog has run out of lives.
- [ ] Write a `succeed1` procedure that executes when the frog completes Level 1, and then add this procedure to the frog.

Lay Out and Code the Level 2 Game Page

- [ ] Create the Level 2 game page.
- [ ] Write a `succeed2` procedure that executes when the frog completes Level 2, and then add this procedure to the frog.

Debug and Enhance

- [ ] Save, test, and debug.
- [ ] Enhance.

Key Ideas

Adding levels, duplicating pages, transitioning between pages, using the keyboard to control a character in four directions, nesting procedures, creating a project variable to track lives across levels, setting speeds, collisions and conditionals, universal color conditional, changing character shape according to health.