|  |
| --- |
|  |

CIS 170-70: C Programming

EXAM Project Plan: PS1 Designer

Spencer White

11/15/2020

|  |
| --- |
|  |

# Table of Contents

[Table of Contents 2](#_Toc56237265)

[Program / Game Concept 3](#_Toc56237266)

[Key Concepts 3](#_Toc56237267)

[Breakdown 4](#_Toc56237268)

[int main() 4](#_Toc56237269)

[void createNewPrompt() 5](#_Toc56237270)

[void editExistingPrompt() 6](#_Toc56237271)

[void switchBashPrompts() 7](#_Toc56237272)

[void understandingPS1() 7](#_Toc56237273)

[void understandingSpecialPromptVariables() 7](#_Toc56237274)

[void howToSetupPS1() 8](#_Toc56237275)

[void generateRandomPrompt() 8](#_Toc56237276)

# Program / Game Concept

This program will focus on teaching users about PS1 prompts and allow them to create, edit, delete and store them. The user will learn about and be able to select special prompt variable characters to easily add features like the date, current working directory, colors, and custom content or phrases! The user can save prompts they create, select and switch to different prompts from their saved list, and when ready, retrieve instructions on how to enable and use their prompts (on MacOS). The program will also include a feature to generate random prompts, to help inspire the user.

# Key Concepts

1. I/O User Input (Requires Validation)
   1. Edit Prompt Menu – CHAR
   2. Add Content Menu – CHAR
   3. Add Feature Menu – CHAR
   4. Add Custom Text – STRING
   5. Remove Content Menu – CHAR
   6. Switch Feature Menu Part One – CHAR
   7. Switch Feature Menu Part Two – CHAR
   8. Color Handling Menu – CHAR
   9. Add Color Menu Part One – CHAR
   10. Add Color Menu Part Two – CHAR
   11. Remove Color Menu – CHAR
   12. Select Prompt Menu
2. I/O Output
   1. printf() and puts() when rendering menus and presenting results of actions taken by user to update prompt
3. File I/O
   1. Save prompts to file
4. Loops
   1. Do While
      1. All menus defined under "User Input” above
   2. For
      1. Used to loop a number of times within generateRandomFeatures() and generateRandomColors() to create the random prompt.
   3. While
      1. Used to clear the buffer after user input
5. Functions – See detailed function description below under “Breakdown”
6. Conditionals/Control Structures
   1. Switch Statements – for all menus defined under “User Input” above
   2. IF, Else If, Else Statements – used to determine if certain actions should be taken e.g. color can only be removed from a component with color

# Breakdown

1. Data Types
   1. Integers/Constants
      1. numOfFeatures - Number of features to use when generating random prompts
      2. numOfColors - Number of colors to use when generating random prompts
   2. Characters
      1. All menu that consume chars as indicated under User Input below.
         * 1. editPromptMenuChoice
           2. addContentMenuChoice
           3. addFeatureMenuChoice
           4. removeContentMenuChoice
           5. switchFeatureMenuPartOneChoice
           6. switchFeatureMenuPartTwoChoice
           7. colorHandlingMenuChoice
           8. addColorMenuPartOneChoice
           9. addColorMenuPartTwoChoice
           10. removeColorMenuChoice
           11. selectPromptMenuChoice
   3. Strings/Arrays
      1. Each prompt e.g. promptOne = '**[**\u**@**\h:\w**]**\$ '
      2. animals – used with generateRandomFeatures() to add random value from array to prompt
      3. adjectives – used with generateRandomFeatures() to add random value from array to prompt
      4. specialFeatures - used with generateRandomFeatures() to add random value from array to prompt
      5. colors – used with generateRandomColors() to add random value from array to feature of prompt
   4. Structs
      1. Stored prompts file e.g. { promptOne: '**[**\u**@**\h:\w**]**\$ '}
      2. randomOptions – used with generateRandomFeatures() and generateRandomColors() to add random values from arrays to prompt e.g. {animals: [], adjectives: [], specialFeatures: [], colors: []}
   5. Pointers
      1. Used to handle arrays passed to functions and modification of the original value

## int main()

When main() is called, displays *Main Menu*. Based on the selection choice of the user, one of the described functions will be called.

*Main Menu:*

1. Create A New Bash Prompt – CALLS - createNewPrompt()
2. Edit An Existing Bash Prompt – CALLS – editExistingPrompt()
3. Switch Bash Prompt – CALLS – howToSwitchBashPrompt()
4. Understanding PS1 – CALLS – understandingPS1()
5. Understanding PS1 Special Variables – CALLS – understandingSpecialPromptVariables()
6. Setup Prompt on MacOS – CALLS – howToSetupPS1()
7. Generate Random Prompt – CALLS – generateRandomPrompt()
8. Quit (Exits Program)

## void createNewPrompt()

Displays the *Edit Prompt Menu*:

*Edit Prompt Menu*:

1. Add Content – CALLS – addContentMenu() – displays *Add Content Menu*

*Add Content Menu*

* + 1. Add Default Feature (Append Only) – displays *Add Feature Menu*

*Add Feature Menu* (On select, adds feature to prompt and displays updated value)

* + - 1. Date
      2. Hostname
      3. Jobs
      4. Shell Terminal Basename
      5. 24 Hour Time
      6. 12 Hour Time
      7. Username
      8. Bash Version
      9. Current Working Directory
      10. Quit (Returns from Sub Menu)
    1. Add Custom Text (Append Only)
       1. receiveInputFromUser() – reads input and appends to current bash prompt
    2. Quit (Returns from Sub Menu)

1. Remove Content – CALLS – removeContentMenu() – displays *Remove Content Menu*

*Remove Content Menu* (On select, deletes the feature from prompt and displays updated value)

* 1. Prompt Part One
  2. Prompt Part Two
  3. Prompt Part Three
  4. Prompt N
  5. Quit (Returns from Sub Menu)

1. Move Content – CALLS – moveContentMenu() – displays *Switch Feature Menu Part One*

*Switch Feature Menu Part One* (On select, takes this feature as the one to move and loads *Switch Feature Menu Part Two*)

* 1. Prompt Part One
  2. Prompt Part Two
  3. Prompt Part Three
  4. Prompt N
  5. Quit (Returns from Sub Menu)

*Switch Feature Menu Part Two* (On select, takes the first feature and inserts at location, pushing later features down one position.

1. Prompt Part One
2. Prompt Part Two
3. Prompt Part Three
4. Prompt N
5. Quit (Returns from Sub Menu)
6. Color Content – CALLS – colorHandlingMenu() – displays *Color Handling Menu*

*Color Handling Menu*

* 1. Add Color – CALLS – addColorMenu() – displays *Add Color Menu Part One*

*Add Color Menu Part One* (On select, takes this feature as the one to add color to and loads *Add Color Menu Part Two*)

1. Prompt Part One
2. Prompt Part Two
3. Prompt Part Three
4. Prompt N
5. Quit (Returns from Sub Menu)

*Add Color Menu Part Two* (On select, adds this color to the feature selected above)

1. Red
2. Green
3. Yellow
4. Blue
5. Purple
6. Turqoise
7. Quit (Returns from Sub Menu)
   1. Remove Color – CALLS – removeColorMenu() – displays *Remove Color Menu*

*Remove Color Menu* (On select, removes color from this feature, IF color is currently attached)

1. Prompt Part One
2. Prompt Part Two
3. Prompt Part Three
4. Prompt N
5. Quit (Returns from Sub Menu)
6. Save Prompt to File – CALLS - writePromptToFile() – Appends current prompt to saved\_prompts.txt
7. Quit (Returns from Sub Menu)

## void editExistingPrompt()

When editExistingPrompt() is selected, calls readPromptFromFile(). IF no prompts exists, a error message is show. If prompts do exist, the user is then asked to select a prompt. After selecting a prompt, the *Edit Prompt Menu* is displayed.

*Select Prompt Menu:*

1. Prompt One
2. Prompt Two
3. Prompt Three
4. Prompt N
5. Quit (Returns form Sub Menu)

*Edit Prompt Menu:* (Shortened below, see above for more details)

1. Add Content
2. Remove Content
3. Move Content
4. Color Content
5. Save Prompt to File
6. Quit (Returns form Sub Menu)

## void howToSwitchBashPrompts()

When howToSwitchBashPrompts() is selected, the user is asked to select a prompt loaded from their prompts file. After selecting a prompt, the howToSetupPS1() is called.

*Select Prompt Menu:*

1. Prompt One
2. Prompt Two
3. Prompt Three
4. Prompt N
5. Quit (Returns form Sub Menu)

## void understandingPS1()

When understandingPS1 () is called, the function will print detailed information about PS1. Resources used as reference:

* + 1. <https://www.linuxnix.com/linuxunix-shell-ps1-prompt-explained-in-detail/>
    2. <https://ss64.com/bash/syntax-prompt.html>
    3. <https://linuxconfig.org/bash-prompt-basics>

## void understandingSpecialPromptVariables()

When understandingSpecialPromptVariables () is called, displays all the special variables, their effect on the prompt, and an example, e.g.:

Special Prompt Variables:

* 1. Current Working Director – CODE: \w – displays the users current working directory e.g. /Users/userOne/Desktop
  2. Username – CODE: \u – displays the username of the current user e.g. spark10

## void howToSetupPS1()

When howToSetupPS1 () is called, displays instructions on how to setup PS1 in your terminal:

Option 1:

* 1. Copy this value export PS1=”SOME VALUE” using cmd + v.
  2. Open your terminal
  3. Run these commands:
     1. cd ~/
     2. vi .bash\_profiile
     3. Type i
     4. Paste using cmd + v
     5. Hit escape
     6. Type :wq and hit enter
  4. Close your terminal and re-open to see the changes

Option 2:

* 1. Open your terminal
  2. Run this command:
     1. echo export PS1=”SOME VALUE” >> ~/.bash\_profile
  3. Close your terminal and re-open to see the changes

## void generateRandomPrompt()

When generateRandomPrompt () is called, the function will:

1. Call generateRandomFeatures() – which generates a random prompt composed of a random number of features indicated below:
   1. Date
   2. Hostname
   3. Jobs
   4. Shell Terminal Basename
   5. 24 Hour Time
   6. 12 Hour Time
   7. Username
   8. Bash Version
   9. Current Working Directory
   10. Custom Phrases – Combination of adjective[] and animal[]
2. Call generateRandomColors() – which will use random colors to color a random number of the features in the prompt returned by generateRandomFeatures()
3. Call displayPromptResults() – while will display the prompt, the features that make it up, and the color codes used to color the features
4. Call savePrompt() – displays *Save Prompt Menu*

*Save Prompt Menu:*

* 1. Save Random Prompt – CALLS – writePromptToFile() – Appends current prompt to saved\_prompts.txt
  2. Quit (Returns form Sub Menu) and randomPrompt is lost