# Assignment – Writing iOS Apps with UIKit

Curriculum Expectations: B1.4, B1.6, B2.1, B2.2, B2.3, B3.1, B3.2, B3.3, B3.4

The purpose of this assignment is to develop a beginning understanding of how to :

* write a program with a visual user interface
* organize a program using design patterns (in this case, Model-View-Controller)
* build a program that meets a list of specifications
* extend and use a test plan to evaluate program correctness

## Core Specifications

The completed calculator program shall:

1. display…
   1. provided values
2. from a *cleared* state, display the…
   1. product of the provided and existing computed value
   2. quotient of the provided and existing computed value
   3. sum of the provided and existing computed
   4. difference of the provided and existing computed value
   5. percentage of a provided value, expressed as a decimal
3. with an existing computed value, then determine the…
   1. product given another provided value
   2. quotient given another provided value
   3. sum given another provided value
   4. difference given another provided value
   5. percentage, expressed as a decimal
4. permit the toggling of a computed or provided value from negative to positive values at any time
5. perform all of the above operations with integer or decimal values

## Extended Specifications (Bonus)

F. The calculator displays a visual cue to indicate the currently selected operation.

Programing:

Calculator Plans

A. Inquiry Plan

Required outlets? Required actions? What state should the model track? What functions does the model need?

*labelDisplay onePressed computedValue : Double addToNewValue(digit : string)*

*twoPressed providedValue : String? multiply*

*threePressed operation : Operation? divide*

*fourPressed overflow : Double updateState*

*fivePressed equals*

*sixPressed makeProvidedValueCompletedValue*

*sevenPressed makeComputedValueProvidedValue*

*eightPressed plusminus*

*ninePressed decimal*

*zeroPressed percentage*

*percentage add*

*decimal subtract*

*plusminus clear*

*equals makeOverflowProvidedValue*

*multiply makeProvidedValueOverflow*

*divide*

*add*

*subtract*

*clear*

B. Test Plan (A plan used to make sure the code works in every way needed)

Specifications

The completed calculator program shall:

1. display…
   1. provided values
2. from a *cleared* state, display the…
   1. product of the provided and existing computed value
   2. quotient of the provided and existing computed value
   3. sum of the provided and existing computed
   4. difference of the provided and existing computed value
   5. percentage of a provided value, expressed as a decimal
3. with an existing computed value, then determine the…
   1. product given another provided value
   2. quotient given another provided value
   3. sum given another provided value
   4. difference given another provided value
   5. percentage, expressed as a decimal
4. permit the toggling of a computed or provided value from negative to positive values at any time
5. perform all of the above operations with integer or decimal value

| Test Cases | | |
| --- | --- | --- |
| **For specification(s)…** | **Input (button sequences)** | **Expected Output (in calculator's label)** |
| A1 | **C37** | **37** |
| A1 | **C37.1** | **37.1** |
| A1, D | **C37.1±** | **-37.1** |
| A1, D | **C37.1±±** | **37.1** |
| B1 | **C3×4=** | **12** |
| B1 | **C3.1×4=** | **12.4** |
| B1 | **C±3×4=** | **-12** |
| D | **C±3×4=±** | **12** |
| D | **C±3×4=±±** | **-12** |
| B2 | **C30**÷**3=** | **10** |
| B2 | **C30**÷**4=** | **7.5** |
| B2 | **C1**÷**3=** | **0.3333333333** (to limit of label) |
| B2, E | **C6.6**÷2.2= | **3** |
| B2, E, D | **C±4.8**÷1.2= | **-4** |
| B2, E, D | **C±4.8**÷**±1.2=** | **4** |
| B3, | **C10+4** | **14** |
| B3, E | **C10.6+4.2=** | **14.8** |
| B3, E, D | **C±10.6+4.2=** | **-6.4** |
| B3, E, D | **C±10.6+±4.2=** | **-14.8** |
| B4, D | **C4-2=** | **2** |
| B4, E, D | **C±8.6-4.3=** | **-12.9** |
| B4, E, D | **C±8.6-±4.3=** | **-4.3** |
| B4, E, D | **C4.4-2=±±** | **2.4** |
| B1, CD | **C3×4=-4=** | **8** |
| B2, CC | **C50**÷**5=+10=** | **20** |
| B3, CB | **C24+4=**÷5= | **4** |
| B4, CA | **C3-6=×6=** | **-18** |
| B1, CC, E | **C3×5.2=+23=** | **39.6** |
| B2, CD, E | **C12.6**÷6**=-2=** | **0.1** |
| B3, CA | **C5+6=×**5= | **55** |
| B4, CB, E | **C15.5-5=**÷**5=** | **2.1** |
| B5 | **C34%=** | **0.34** |
| B5, D | **C34%=±** | **-0.34** |
| B5, D | **C3%×5±=** | **-0.15** |
| B5 | **C12%÷4%=** | **0.16** |
| B3, CE | **C5+6=%=** | **0.11** |
| B5, E | **C2+8.2=%=** | **0.012** |

## Part A – Making an Implementation Plan (Inquiry)



*Complete the plan to separate concerns (user interface in the view, state kept in the model, connect logic in controller).*

Required outlets? Required actions? What state should the model track? What functions does the model need?

*labelDisplay onePressed computedValue : Double addToNewValue(digit : string)*

*twoPressed providedValue : String? multiply*

*operation : Operation? divide*

*updateState*

*equals*

*makeProvidedValueCompletedValue*

*clear*

| Evaluation Criteria |  |  |  |
| --- | --- | --- | --- |
| **Necessary outlets, actions, state information, and functions identified** | None | 0 1 2 3 4 5 6 7 8 9 10 | All |

## Part B – Making a Test Plan (Inquiry)

*Complete the plan by devising a test case that verifies your program meets each specification given on page one.*

| Test Cases | | |
| --- | --- | --- |
| **For specification(s)…** | **Input (button sequences)** | **Expected Output (in calculator's label)** |
| A1 | **C37** | **37** |
| A1 | **C37.1** | **37.1** |
| A1, D | **C37.1±** | **-37.1** |
| A1, D | **C37.1±±** | **37.1** |
| B1 | **C3×4=** | **12** |
| B1 | **C3.1×4=** | **12.4** |
| B1 | **C±3×4=** | **-12** |
| D | **C±3×4=±** | **12** |
| D | **C±3×4=±±** | **-12** |
| B2 | **C30**÷**3=** | **10** |
| B2 | **C30**÷**4=** | **7.5** |
| B2 | **C1**÷**3=** | **0.3333333333** (to limit of label) |
|  |  |  |

| Evaluation Criteria |  |  |  |
| --- | --- | --- | --- |
| **Test plan will verify that program operates correctly for all given specifications** | Not at all | 0 1 2 3 4 5 6 7 8 9 10    11 12 13 14 15 16 17 18 19 20 | Comprehensively |

## Part B – Product

**Application**

| Evaluation Criteria |  |  |  |
| --- | --- | --- | --- |
| **Program passes a comprehensive test plan** | None | 0 1 2 3 4 5 6 7 8 9 10   11 12 13 14 15 16 17 18 19 20 | All |
| **Adherence of MVC design pattern and use of functions where possible to avoid repeated code** | Not at all | 0 1 2 3 4 5 6 7 8 9 10 | In all cases |

## Communication

| Evaluation Criteria |  |  |  |
| --- | --- | --- | --- |
| **Human-readable code: good naming, use of whitespace, comments to describe intent** | Not at all | 0 1 2 3 4 5 6 7 8 9 10   11 12 13 14 15 | Extremely readable |
| **Correct use of source control to demonstrate evolution of work** | Not at all | 0 1 2 3 4 5 6 7 8 9 10   11 12 13 14 15 | Regular commits with descriptive commit messages |