+
JMJ

Anthony Ranieri

Test Plan (PROBLEM 1)

☑ ask user for starting date [✓ Y] N

☑ convert string (starting date) to integers [✓ Y] N

☑ change month of starting date to days [✓ Y] N

☑ ask user for ending date [✓ Y] N

☑ convert string (ending date) to integers [✓ Y] N

☑ change month of ending date to days. [✓ Y] N

☑ get amount of days between the two dates. [✓ Y] N

☑ add days to the amount of days for leap years [✓ Y] N

☑ output amount of days in total. [✓ Y] N

+
JMJ

Anthony Raneri Test Plans (PROBLEM2) A

- ☑ put in srand() function right away for randomizing. [✓ Y] N
- ☑ ask user for computer to play in 'smart' or 'stupid' mode. [✓ Y] N
- ☑ ask user who gets first turn. [✓ Y] N
- ☑ ask user to initialize the size of the pile. [✓ Y] N
- ☑ ~~#~~ code game for stupid mode and computer goes first. [✓ Y] N
  - ☑ while loop until pile goes away. [✓ Y] N
  - ☑ computer picks a random number. [✓ Y] N
  - ☑ user picks a number. [✓ Y] N
  - ☑ size of pile dwindles as computer & user takes marbles [✓ Y] N
  - ☑ if user picks amount of marbles greater the half of the pile or less than one, tell them to repick. [✓ Y] N
- ☑ code game for stupid mode and user goes first. [✓ Y] N
  - ☑ while loop until pile goes away. [✓ Y] N
  - ☑ user picks a number. [✓ Y] N
  - ☑ if user pick is not valid, tell them to repick. [✓ Y] N
  - ☑ computer picks random number. [✓ Y] N
  - ☑ size of pile dwindles after each pick [✓ Y] N
- ☑ code game for smart mode and computer goes first. [✓ Y] N
  - ☑ while loop until pile goes away. [✓ Y] N
  - ☑ computer gets random number [✓ Y] N
    - ☑ → program it after random numbers to square it and subtract one. [✓ Y] N
  - ☑ computer ~~takes~~ picks number [✓ Y] N
  - ☑ user picks number [✓ Y] N
  - ☑ if user pick is not valid, tell them to repick [✓ Y] N

+
3/13

- ☑ size of pile dwindles after each pick ✓ (PROBLEM a) B
- ☑ if size of pile is less or equal to 5
  ✓ computer has to pick 1 [✓ Y] N
- ☑ ~~code~~ program when computer in smart mode and user picks first [✓ Y] N
  - ☑ while loop until pile goes away. [✓ Y] N
  - ☑ user picks ~~first~~ number. [✓ Y] N
  - ☑ user pick not valid, tell them to repick [✓ Y] N
  - ☑ computer gets random number (a small number) [✓ Y] N
  - ☑ ↳ then square it and subtract one [✓ Y] N
  - ☑ ↳ if too big goes in while loop till it
    gets a small enough number [✓ Y] N
  - ☑ size of pile dwindles with each pick. [✓ Y] N
- ☑ outputs whenever computer wins [✓ Y] N
- ☑ outputs whenever user wins [✓ Y] N

- ☑ during <u>each</u> mode & turn setting [✓ Y] N

✝ JMJ

Anthony Ranieri Test Plans (PROBLEM3) A

- ☑ ask user for an integer at ~~to~~ most 10 digits. [✓ Y] N
- ☑ set user input to a variable to use later. [✓ Y] N
- ☑ set 3 int variables to the value zero. [✓ Y] N
- ☑ while loop for length of integer. [✓ Y] N
- ☑
    - ☑ get modulos of the integer [✓ Y] N
    - ☑ set it to one of your variables [✓ Y] N
    - ☑ divide integer by 10 [✓ Y] N
    - ~~□ start loop over again~~ ~~Y N~~
    - ☑ get modulos of new integer [✓ Y] N
    - ☑ set variable to this one [✓ Y] N
    - ☑ divide new integer by 10 [✓ Y] N
    - ☑ that is new integer [✓ Y] N
    - ☑ start loop again [✓ Y] N
- ☑ now we have a second set of numbers from that loop [✓ Y] N
    - ☑ set a new while loop [✓ Y] N
    - ☑ for ~~[illegible]~~ digits in new set/repeat [✓ Y] N
    - ☑ get modulos of new set [✓ Y] N
    - ~~□ [illegible]~~
    - ~~□ add~~
    - ☑ times that number by 2 [✓ Y] N
        - ☑ while loop that gets modulos of that number [✓ Y] N
        - ☑ add two digits together [✓ Y] N
    - ☑ divide new set by 2 [✓ Y] N

last set

†
JMJ

(PROBLEM 3) B

☑ if sum of first set and last set is multiple of 10 → (T/F)

☑ output valid & check digit

☑ if not multiple of 10 ⟶

☑ output not valid

[✓ Y] N
[✓ Y] N
[✓ Y] N
[✓ Y] N