

CompSci 94

Study Guide For Exam1

Exam1 on September 26, 2024



Prof. Susan Rodger

Exam 1 Date

- Exam 1 is Thursday, Sept 26
 - Old tests are on course website, Resources tab
 - See them on exam's date with problems marked out (we have not done if, loops and written functions yet)

Exam Logistics

- Exam is on paper
- Thursday, Sept 26, regular class time
 - More time if you get accommodations
 - Should have gotten email from me
- The exam is your own work
- Do not talk about the exam with anyone until it is handed back
- See the Exam1 reference sheet
 - Alice snapshots of procedure names provided on the exam

Exam Topics - Alice

- Alice Videos on warpwire
 - 2.x, 3.x, 4.1.0-4.1.6
- Setup, camera markers, invisible object markers
- Built-in procedures and functions
- Built-in properties: vehicle, opacity, height, etc
- Do in order, Do together
- Write a procedure with parameters
- Use procedure with arguments

Best Way to Study for Exam

- Study Lecture notes, watch video again
- Study Classwork
 - Can you write a procedure on paper or type in file?
 - Try to recreate a classwork or write on paper
- Old exams are available on course web page
 - See “Old Tests” link (on resources tab)
 - **Practice writing methods on paper**
 - Most important – practice writing code

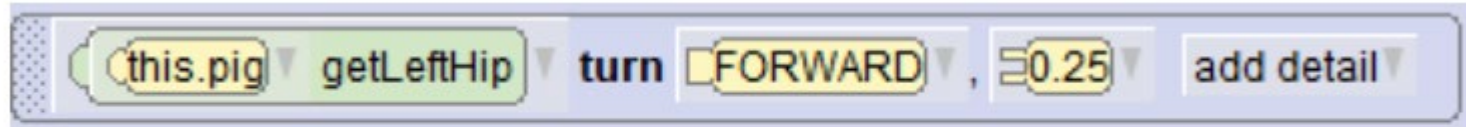
Old exams

- On resources tab on course web page
- Fall 2022, Fall 2021, Fall 2020, Fall 2019 and Spring 2019 – most like your exam
- Fall 2018, Spring 2018 – Alice 3 (material in different order)
- **See list of questions to study, ignore other ones**
- Practice writing code on paper

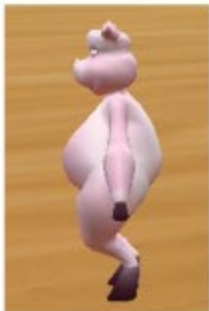
Some Practice questions

Problem 1

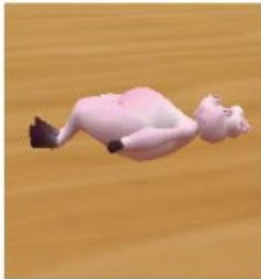
- Consider the following Alice code and the pig is standing straight up as shown with Start in the figure on the left below. Which figure A)-D) is where the pig will be after this line of code is executed?



Start



A)



B)



C)

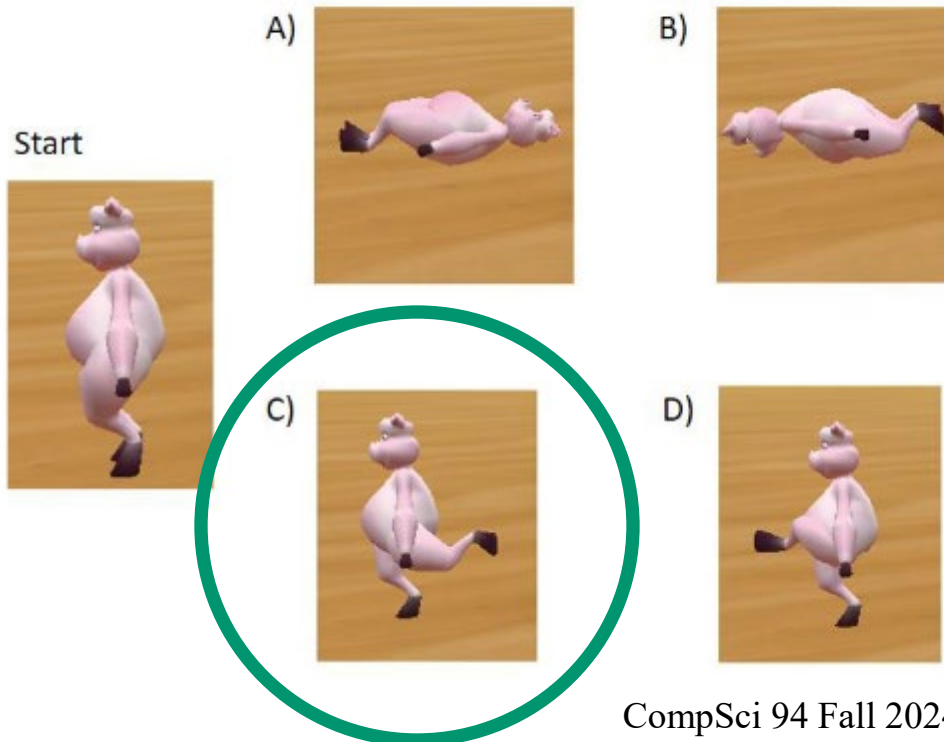
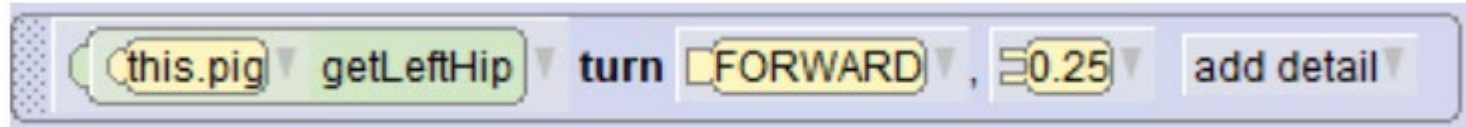


D)



Problem 1

- Consider the following Alice code and the pig is standing straight up as shown with Start in the figure on the left below. Which figure A)-D) is where the pig will be after this line of code is executed?



You should practice writing code

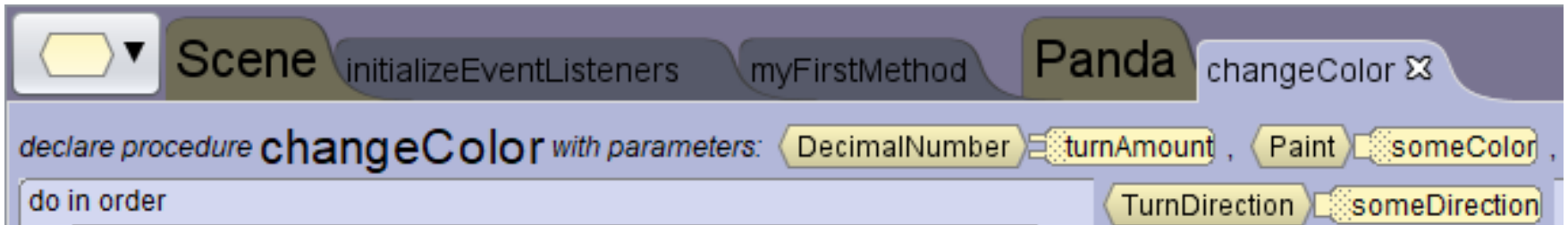
- Practice writing code from classworks and old exams

Problem 2

Write **panda** Procedure **changeColor**

- This procedure has **three parameters**
 - One parameter of type **Decimal** named **turnAmount**
 - One parameter of type **Paint** named **someColor**
 - One parameter of type **TurnDirection** named **someDirection**.
- When called, taking 3 seconds total, the panda turns around the **turnAmount** in the direction **someDirection** while at the same time changing to the color **someColor**.

Write the procedure changeColor



Solution

The image shows a Scratch code editor with a 'Panda' object selected. A procedure named 'changeColor' is being edited. The procedure has two parameters: 'turnAmount' (DecimalNumber) and 'someColor' (Paint). The code inside the procedure is as follows:

```
declare procedure changeColor with parameters: DecimalNumber turnAmount, Paint someColor, TurnDirection someDirection
do in order
  do together
    this setPaint someColor, duration 3.0 add detail
    this turn someDirection, turnAmount, duration 3.0 add detail
```

The code is written in a block-based style, with 'this' representing the object, 'setPaint' and 'turn' being the actions, and 'duration 3.0' and 'add detail' being the timing and detail settings.

Give the two calls to `changeColor`

- Give the call that has the panda turn right twice while turning Blue
- Give the call that has the panda turn left 1.5 times while turning Green

Give the two calls to `changeColor`

- Give the call that has the panda turn right twice while turning Blue

```
this.panda.changeColor(turnAmount: 2.0, someColor: BLUE, someDirection: RIGHT)
```

- Give the call that has the panda turn left 1.5 times while turning Green

```
this.panda.changeColor(turnAmount: 1.5, someColor: GREEN, someDirection: LEFT)
```

Problem 3:

Write **Bunny** Procedure **funJumping**

- This procedure has **four parameters**
 - One parameter of type **Decimal** named **opValue**
 - Two parameters of type **Paint** named **color1, color2**
 - One parameter of type **Sdisc** named **someDisc**
- Before called, the bunny is standing on a disc that will be passed as an argument



funJumping story(cont)

- The disc moves up 1 and back down to the ground carrying the bunny up and down with it. As the disc moves up it changes its color to color1 and the bunny changes its color to color2.
- Next the bunny changes its opacity to opValue
- The disc moves up 1 and back down again with the bunny
- Then instantly, the bunny turns back to its original color, the bunny is no longer faded and the disc disappears.

Write the procedure funJumping

declare procedure **funJumping** with parameters: **DecimalNumber** **opValue** , **Paint** **color1** ,
do in order
do in order **Paint** **color2** , **SDisc** **someDisc**

A solution

The image shows a Scratch script for an object named 'Bunny'. The script is titled 'funJumping' and has two parameters: 'opValue' (DecimalNumber) and 'color1' (Paint). The script is organized into nested 'do in order' blocks. The outer block contains a 'Paint' block for 'color2' and an 'SDisc' block for 'someDisc'. The inner 'do in order' block contains a 'this' block for 'setVehicle' with 'someDisc', followed by a 'do together' block. The 'do together' block contains three parallel actions: 'someDisc' move UP by 1.0, 'someDisc' setPaint to 'color1', and 'this' setPaint to 'color2'. After the 'do together' block, the script continues with a sequence of actions: 'someDisc' move DOWN by 1.0, 'this' setOpacity to 'opValue' with duration 0.0, 'someDisc' move UP by 1.0, 'someDisc' move DOWN by 1.0, 'this' setPaint to 'WHITE' with duration 0.0, 'this' setOpacity to 1.0 with duration 0.0, and finally 'someDisc' setOpacity to 0.0 with duration 0.0. Each action block has an 'add detail' button.

Scene initializeEventListeners myFirstMethod **Bunny** funJumping ✕

declare procedure **funJumping** with parameters: DecimalNumber **opValue** , Paint **color1** ,

do in order

Paint **color2** , SDisc **someDisc**

do in order

- this **setVehicle** **someDisc**
- do together
 - someDisc** move UP , 1.0 add detail
 - someDisc** setPaint **color1** add detail
 - this **setPaint** **color2** add detail
- someDisc** move DOWN , 1.0 add detail
- this **setOpacity** **opValue** , duration 0.0 add detail
- someDisc** move UP , 1.0 add detail
- someDisc** move DOWN , 1.0 add detail
- this **setPaint** **WHITE** , duration 0.0 add detail
- this **setOpacity** 1.0 , duration 0.0 add detail
- someDisc** **setOpacity** 0.0 , duration 0.0 add detail

Now you should look at some of
the old exams