CS 111 Final Project  
Self Assessment

Your group should fill out one copy of this form and include it with your assignment when you turn it in.

# Academic integrity statement

By turning in this assignment, you certify that:

* The information below, including the listing of who did what, is accurate so far as you understand it
* You have not included any code you did not write yourselves, other than the code handed out as part of the original assignment, or examples used in class
* If you used example code from class, you did not count it in your methods, procedures, types, etc., below.
* You have not shared your code with anyone not in your group

# Group

*Who’s in your group?*

1. Matthew Poelsterl
2. Fatima Malik
3. Victor Achuthan

# Goals

*Say a few words about what you wanted the game to be like. Note that if you just wanted to write some code so you could get an good grade on the project, it’s fine to admit that.*

Our goal was to simulate waking up in the morning, going through a typical morning routine, and then leaving for the day. The idea of the game was to successfully leave the house after completing all the necessary tasks like dressing up, engaging with pets, feeding yourself, and putting stuff in your bag.

# Lessons learned

*What went right?*

We were able to break our program down into smaller pieces, code those small parts one by one and use inheritance effectively, which taught us the benefits of objected-oriented programming.

*What went wrong?*

We originally wanted to create an environment that had more conditions and had an actual win/lose component with a timer, but decided after we started that it was too ambitious, and scaled back our approach.

*What do you wish you knew when you started?*

It was confusing at first to understand the relationship between all the types (inheritance).

# Annoying grading bookkeeping

## Types (8)

*What are the types you added, and what are they for?*

1. Organism - parent type of animal and person to handle eating
2. Animal - subtype of organism, allows for petting an animal
3. Food - represents a prop that can be eaten
4. Utensil - prop needed to eat certain food
5. Key - prop needed to unlock a door
6. Clothing - represents clothes that can be taken on and off with a slot system
7. Phone - type for a phone that can give weather, temperature, and time
8. Bag - portable container with a capacity limit

## Fields (18)

*What are the fields you added, what types did you add them to, and what are they for?*

1. Fields of organism:

1. hunger - specifies level of hunger of organism

2. Fields of animal:

1. Friendliness - specifies level of friendliness for animal

3. Fields of food:

1. fillingness - specifies how filling food item is for organism

2. for-human? - specifies (t/f) if food item if for human

3. needs-spoon? - specifies (t/f) if food item should be consumed with spoon

4. needs-knife? - specifies (t/f) if food item should be consumed with knife

4. Fields of utensil:

1. kind - keeps track of the type of utensil

5. Fields of key:

1. color - specifies color of utensil

6. Fields of clothing:

1. kind - specifies the type of clothing (hat, glove, etc)

2. warmth - specifies level of warmth article of clothing provides

7. Fields of phone:

1. temperature - specifies the temperature outdoors given by phone's app

2. time - specifies the time given by the phone

3. weather -specifies the weather (snowy/rainy) given by phone's weather app

8. Fields of bag:

1. openZip? - specifies (t/f) if bag zipper is open

2. capacity - specifies the capacity of bag i.e number of items bag can hold.

1. Additional fields for person:

1. outfit - holds clothes, separate from contents to facilitate slot system

2. Additional fields for door:

1. locked? -specifies (t/f) if door is locked

2. open? - Specifies (t/f) if door can be opened

## Procedures (9)

*What are the procedures you added or significantly modified from their original form, and what are they for?*

1. new-food - creates a new food object.

2. new-utensil - creates a new utensil object.

3. new-key - creates a new key object.

4. new-clothing - creates a new clothing object.

5. new-animal - creates a new animal object .

6. new-bag - creates a new bag object.

7. new-phone - creates a new phone object .

8. outfit - creates a new food object.

1. Additional Procedures for Door:

1. new-door - creates a new door object.

## Methods (15)

*What are the methods you added or significantly modified from their original form, what types were they added to, and what are they for? Note that if you have three different methods for the same generic procedure, list each one separately.*

1. Methods for organism:

1. feed - checks if organism can eat the thing, if so then it is removed from location and fullness of organism is updated

2. can-eat? - checks if the input object is appropriate for the particular organism to eat, returns t/f

2. Methods for animal:

1. pet - when called, prints animal's reaction to being pet, based on the friendliness field of animal. Also, increases the friendliness of the animal by adding one and multiplying by two.

3. Methods for clothing:

1. don - Puts on a clothing item; checks if clothing item is already being worn, otherwise, the clothing item is moved onto the person.

2. doff - Takes of clothing item; checks if item of clothing is not being worn, otherwise, removes the clothing item off of the person.

4. Methods for phone:

1. check-weather - prints the outside weather

2. check-time - prints the time for the first appointment of the day

3. check-temperature - prints the temperature outside.

5. Methods for bag:

1. zipOpen-bag - Opens bag (Doesn't count toward requirement total, added in list for clarity)

2. zipClose-bag - closes bag (Doesn't count toward requirement total, added in list for clarity)

3. putIn-bag - Checks if bag is open, Checks if bag has reached the capacity limit, if not, adds thing to the bag's contents

4. takeOut-bag - checks if bag is already opened, then takes thing out of bag

1. Additional methods for door:

1. unlock - unlocks door

2. open - opens the door if it's unlocked ((Doesn't count toward requirement total, added in list for clarity)

2. Additional methods for person:

1. can-eat? - checks if a person has all the items necessary to consume the food item. For example, if the food needs a spoon to be eaten, method checks if a spoon is in the person's contents.

## Total stuff we built

*Write the total number of items listed above.*

50

# Who did what

For each person in your group, list the types, procedures, and methods that they implemented, and anything else they did as part of the project.

Matthew: implemented organism, animal, clothing, and their associated methods

Fatima: implemented key, utensil, and associated methods with the door; wrote walkthrough

Victor: implemented bag and phone; wrote up descriptions for this self assessment