# **Social Psychology Chapter 6**

**Emotions**

### **Characterizing emotion**

* A specific response both psychological and physiological:
* Helps people meet goals (including social goals)
* Last a few seconds (don’t usually last more than 20 sec)
* Emotion is cross-cultural / everyone experiences the same emotions
* Emotion can be a tool to get certain needs met

### **Darwin's Hypothesis**

* Emotions are universal
* All humans have the same facial muscles
* Since humans share an evolutionary history with other mammals most recently primates, our emotionally expressive behaviors should resemble those of other species
* Blind people still demonstrate the same facial expressions

The universality of facial expressions

* Facial expressions are recognized cross-culturally
* Vulnerability — the ability to be wounded
* “We have sacrificed authenticity for attachment”

Recognizing facial expressions and emotions

* Most people can others' emotions

### **Case study: Edward Muybridge:**

* Commitment problem— long-term relationships require us to sacrifice for others even when we don't always wish to.
* Shallow commitment —-- surface level emotions/ lack of commitment// detachment/disconnected emotionally
* Deep commitment — during hard times you can develop deeper emotions; and deeper connection
* You have to love yourself first before you can be loved by others

### **Emotions and social relationships**

* Emotions coordinate our interactions with others - ex… playing on a team, reconciling, flirting
* People who know others' emotions and theirs do better in every stage of their lives

#### **Promoting commitment**

* Emotions promote relationship commitment
* Expression of emotion signals a sincere commitment to others
* Emotions motivate us to prioritize the welfare of others
* If your not good at expressing your emotion state you're not good at predicting others emotional states

- Communicating emotion through touch

* With a brief touch of the forearm, participants would communicate different emotions to strangers

### **Knowing our palace in groups**

* Emotions help us find our status within groups
* People rely on emotions to determine their status (pride and anger)
* People see anger as power and status
* “When you have the law on your side argue the law if you don't just pound the table”

### **Emotions and social Cognition**

* Emotions can influence how we process information and make judgments.
* Emotions can influence judgments by being taken as additional information about the judgment
* Duality of hope

#### **Emotions influence perceptions**

* Color based metaphors portraying emotions
* Sadness blue
* Anger red
* Happiness rose colored

#### **Emotions influence reasoning**

* Broaden-and-build hypothesis…
* Positive emotions broaden thoughts and actions, helping people build social resources
* Global local decision task pos— triangle neg—-square

#### **Emotions influence moral judgment**

* Social intuitionist model of moral judgment
* People first have fast, emotional reaction leading to judgment of “right” or “wrong”
* Ex. liberals and conservatives and idea of authority
* Create haves vs. have nots
* 99% vs. 1% setting the populace without actual power against each other

### **Moral foundations**

- Moral foundation theory

* Five evolved, universal moral judgment domains
* cair/harm
* fairness/cheating
* loyalty/betrayal
* Authority/subversion
* Purity/degradation

#### **Knowing What makes us happy**

- Affective forecasting

- Predicting future emotions

- Affective forecasting is often incorrect

- Our image of what something will be like is not always accurate

#### **Immune Neglect**

* Underestimating their capacity to be resilient in responding to difficult life events which leads
* - overestimunate extent to which lifes problems will reduce their personal well-being

### **Focalism**

* Tendency to focus too much on a central aspect of an event
* ex) one college visit

Write down five things you are grateful for

1. I am grateful for access to Food and Water since I know some people don’t have access to it and we all as humans need it to survive
2. I am grateful for Music, because I feel happy when I am able to listen to music and I enjoy exploring different songs and audio art
3. I am grateful for to be young since I gives me a lot of opportunity and I have the ability to explore many different paths
4. I am grateful for my parents since they are alive and well, and have always supported me and allowed me to live how I chose to most of the time
5. I am grateful for books since I like to explore different worlds and learn about anything I wish

import os # Citing os from https://docs.python.org/3/library/os.html

import random # Citing random from https://docs.python.org/3/library/random.html

import time # Citing time from https://docs.python.org/3/library/time.html

# Initialize Lists

subject = []

old\_incorrect\_equation = []

old\_incorrect\_answer = []

#Start Function/Initialize Variables

def start():

global names

global incorrect

global correct

global level

global q\_num

global retry\_amount

correct = 0

incorrect = 0

level = 0

q\_num = 0

retry\_amount = 0

name = input("Hello, what's your name: ")

print(name,", what level would you like to start with?")

level\_dec()

totaltime = round((time.time() - start\_time), 2) # Citing time from https://docs.python.org/3/library/time.html

print("\nYour time is: " +str(totaltime)+ " seconds")

if incorrect == 1:

print("You had", str(incorrect), "incorrect answer.")

else:

print("You had", str(incorrect), "incorrect answers.")

if correct == 1:

print("You had", str(correct), "correct answer!")

else:

print("You had", str(correct), "correct answers!")

final\_score\_percent = round(round((correct/(incorrect+correct)),2)\*100)

print("Your overall score is: " +str(final\_score\_percent)+ "%")

reset()

#Function that gives 3 options after compleating a round 1) restart 2)play incorrect answers 3)quit

def reset():

global retry\_amount

if retry\_amount <= 0:

reset=input("\nWould you like to replay, work on incorrect answers, or quit? \n Press 1 to continue\n Press 2 to retry the last 10 incorrect answers\n Press 3 to stop: ")

if reset == "1":

time.sleep(1) # Citing time from https://docs.python.org/3/library/time.html

os.system("clear") # Citing os from https://docs.python.org/3/library/os.html

start()

elif reset == "2":

retry\_amount += 1

retry\_old\_incorrect()

else:

time.sleep(1) # Citing time from https://docs.python.org/3/library/time.html

os.system("clear") # Citing os from https://docs.python.org/3/library/os.html

print("Thank you for playing!")

elif retry\_amount > 0:

reset=input("\nWould you like to replay or quit? \n Press 1 to continue\n Press 2 to stop: ")

if reset == "1":

time.sleep(1) # Citing time from https://docs.python.org/3/library/time.html

os.system("clear") # Citing os from https://docs.python.org/3/library/os.html

start()

elif reset == "2":

time.sleep(1) # Citing time from https://docs.python.org/3/library/time.html

os.system("clear") # Citing os from https://docs.python.org/3/library/os.html

print("Thank you for playing!")

#Function that decides which level path player takes

def level\_dec():

global level

level\_type = int(input("\n 1) Easy\n 2) Normal\n 3) Difficult\n\n "))

print(" ")

q\_num = int(input("How many questions do you want?(1-20): "))

if level\_type == 1:

level = 1

math1(q\_num)

elif level\_type == 2:

level = 2

math2(q\_num)

elif level\_type == 3:

level = 3

math3(q\_num)

subject.append("h")

else:

print("Please choose an option 1-3")

level\_dec()

#MATH Level 1

def math1(q\_num): # if players choose level 1 of math then they get this function

global start\_time

if q\_num > 20 or q\_num < 1:

print(" ")

q\_num = int(input("Please pick a number between 1-20: "))

math1(q\_num)

else:

start\_time = time.time() # Citing time from https://docs.python.org/3/library/time.html

for i in range(q\_num): # Dictates how many questions person gets based off input

choose\_random = random.randint(1, 2) # Citing random from https://docs.python.org/3/library/random.html

print("\n")

if choose\_random == 1:

add()

elif choose\_random == 2:

subtract()

#MATH Level 2

def math2(q\_num): # if players choose level 2 of math then they get this function

global start\_time

if q\_num > 20 or q\_num < 1:

print(" ")

q\_num = int(input("Please pick a number between 1-20: "))

math2(q\_num)

else:

start\_time = time.time() # Citing time from https://docs.python.org/3/library/time.html

for i in range(q\_num):

choose\_random = random.randint(1, 3) # Citing random from https://docs.python.org/3/library/random.html

print("\n")

if choose\_random == 1:

add()

elif choose\_random == 2:

subtract()

elif choose\_random == 3:

mult()

#MATH Level 3

def math3(q\_num): # if players choose level 3 of math then they get this function

global start\_time

if q\_num > 20 or q\_num < 1:

print(" ")

q\_num = int(input("Please pick a number between 1-20: "))

math3(q\_num)

else:

start\_time = time.time() # Citing time from https://docs.python.org/3/library/time.html

for i in range(q\_num):

choose\_random = random.randint(1, 4) # Citing random from https://docs.python.org/3/library/random.html

print("\n")

if choose\_random == 1:

add()

elif choose\_random == 2:

subtract()

elif choose\_random == 3:

mult()

elif choose\_random == 4:

div()

#Addition function

def add():

global level

global correct

global incorrect

if level == 1:

# Citing random from https://docs.python.org/3/library/random.html

x = random.randint(0, 20)

y = random.randint(0, 20)

# End cited code

elif level == 2:

# Citing random from https://docs.python.org/3/library/random.html

x = random.randint(0, 50)

y = random.randint(0, 50)

# End cited code

elif level == 3:

# Citing random from https://docs.python.org/3/library/random.html

x = random.randint(-100, 100)

y = random.randint(-100, 100)

# End cited code

print("What is",x,"+",y,)

add\_q = int(input(": "))

if add\_q == x+y:

print("Correct")

correct += 1

else:

print("Incorrect")

incorrect += 1

add\_old\_incorrect(x, "+", y, x+y)

#Subtraction function

def subtract():

global correct

global incorrect

global level

if level == 1:

# Citing random from https://docs.python.org/3/library/random.html

x = random.randint(0, 20)

y = random.randint(0, 20)

# End cited code

if y > x:

a = y

b = x

y = b

x = a

elif level == 2:

# Citing random from https://docs.python.org/3/library/random.html

x = random.randint(0, 50)

y = random.randint(0, 50)

# End cited code

elif level == 3:

# Citing random from https://docs.python.org/3/library/random.html

x = random.randint(-100, 100)

y = random.randint(-100, 100)

# End cited code

print("What is",x,"-",y,)

add\_q = int(input(": "))

if add\_q == x-y:

print("Correct")

correct += 1

else:

print("Incorrect")

incorrect += 1

add\_old\_incorrect(x, "-", y, x-y)

#Multipication function

def mult():

global correct

global incorrect

global level

if level == 2:

# Citing random from https://docs.python.org/3/library/random.html

x = random.randint(0, 12)

y = random.randint(0, 12)

# End cited code

elif level == 3:

# Citing random from https://docs.python.org/3/library/random.html

x = random.randint(-20, 20)

y = random.randint(-20, 20)

# End cited code

print("What is",x,"x",y,)

add\_q = int(input(": "))

if add\_q == x\*y:

print("Correct")

correct += 1

else:

print("Incorrect")

incorrect += 1

add\_old\_incorrect(x, "x", y, x\*y)

#Divison function

def div():

global correct

global incorrect

# Citing random from https://docs.python.org/3/library/random.html

x = random.randint(0, 100)

y = random.randint(1, 100)

# End cited code

print("What is", x,"/", y, "? (Round to the nearest hundredth)")

div\_question = int(input(": "))

if div\_question == round(x/y, 2):

print("Correct")

correct += 1

else:

print("Incorrect")

incorrect += 1

add\_old\_incorrect(x, "/", y, round(x/y,2))

def add\_old\_incorrect(x, sign, y, answer): #Adds old x and y values, equation signs, and answers into a list of answers you got incorrect

old\_incorrect\_equations = (str(x), sign, str(y))

old\_incorrect\_answers = (answer)

old\_incorrect\_equation.insert(0, (old\_incorrect\_equations)) #Inserts equation into the beginning of the list

old\_incorrect\_answer.insert(0, (old\_incorrect\_answers)) #Inserts matching answer to beginning of the list

if len(old\_incorrect\_equation) > 10:

del old\_incorrect\_equation[10]

def retry\_old\_incorrect(): #Allows users to retry equations they've gotten incorrect

retry\_equation\_amount = len(old\_incorrect\_equation) #The amount of questions you get is based off the amount of incorrect answers you've given

for i in range(retry\_equation\_amount):

print(" ")

retry\_correct\_answer = old\_incorrect\_answer[0]

retry\_correct\_equation = old\_incorrect\_equation[0]

print(" ".join(retry\_correct\_equation))

retry\_answer = int(input(": "))

if retry\_answer == retry\_correct\_answer:

print("Correct")

else:

print("Incorrect")

print("The correct answer is:", retry\_correct\_answer)

del old\_incorrect\_equation[0]

del old\_incorrect\_answer[0]

reset()

#This is where the program runs

while True:

start()

break