Assignment: The Basics

Goals

- Practice using variables, different data types, operators, print statements and gathering user input.
- Learn about gig work and how, when work is managed by technology, it can both benefit and harm workers.

Part 0: Set-up & Introduction

Get your assignment files set-up and learn about the gig economy.

Download the file workplanner.py. Put the file in your CS111 folder, and open it on VSCode. Create a reflection.txt file as well. Throughout the assignment, there will be reflection questions you will answer here (noted by "*Reflection*").

What is the Gig Economy?

The gig economy refers to a sector of the economy where people work short-term, task- or project-based jobs, often arranged through digital platforms [1]. Common examples include driving for Uber or delivering food with DoorDash. Other terms such as the *sharing economy*, *freelance economy*, or *access economy* are also used to describe this type of work.

Although it feels new, gig work has existed for decades, before the rise of technology. The term "gig" originated in music and entertainment, describing a paid performance by a musician or entertainer [2]. Today, gig work usually refers to work on digital platforms. For example, Uber, a rideshare platform, and DoorDash, a food delivery service. According to The Pew Research Center, 16% of Americans have ever earned money through a digital gig platform, with adults under 30, Hispanic adults and those with lower incomes especially likely to do these jobs [2].

Reflection: How do you see technology influencing the kinds of jobs people have today?

What are the Benefits?

Imagine you're a student balancing classes, homework and extracurricular activities. You don't have a fixed schedule that allows you to commit to a part-time job. Instead, you download DoorDash and use your occasional two-hour blocks of free time during the week to complete delivery orders. Flexibility is a main benefit of gig work.

Gig work also provides opportunities for people with fewer skills, lower education levels or disabilities to earn money, offering greater mobility to disadvantaged groups [3]. These jobs offer greater autonomy, with workers able to choose when and where they work. They also offer additional or supplemental income. Finally, it benefits companies and consumers. For example, DoorDash provides convenience for customers while increasing sales for restaurants.

Your Task

You will be creating a simple program that helps delivery drivers estimate their expected weekly income. Many factors affect how much a driver earns. For example, location, a driver in Minneapolis is likely to receive a higher volume of orders compared to one in Northfield. Your program will account for this information and calculate the driver's estimated weekly income.

Part 1: Getting Started with Variables

Define variables for base pay, location, days and more.

Open workplanner.py. As mentioned before, there are many factors that affect income. Let's start with some fixed variables. Add base pay per order, set to $5.

```python
# Fixed variables go here
base_pay = 5
```

Next, create a variable for the average number of jobs per hour, set to 3.

Moving to variables that depend on the driver's preferences, implement the following (you choose the values):

- Hours available to work
- Fuel efficiency (miles per gallon)
- Miles expected to travel
- Cost of fuel

There will be two variables, days and neighborhoods, that the driver can choose multiple of. These will be stored as lists. We will learn more about lists later, but here is a preview of the syntax with the neighborhoods variable. The driver can choose from neighborhoods labeled A-D.

```python
# Driver set variables go here
base_pay = 5
```

Next, make a variable, set to a list, for days the driver is choosing to work.

Reflection: What other variables might affect income? How?

Part 2: Calculations

Use operators to estimate weekly income.

Calculate the drivers revenue by multiplying base pay, average number of jobs per hour, hours expected to work, the neighborhood multiplier and the day multiplier. **Important**: You are given **pre-calculated multipliers**. These account for how much busier or slower certain areas and days are, and how that affects revenue.

```
# Neighborhood and day multipliers
neighborhood_multiplier = wi.checkList(neighborhoods)
day_multiplier = wi.checkList(days)
```

Then calculate deductions from fuel cost.

Calculate the net income using revenue and deductions. Print this variable.

Use these values to check your work: Jane Doe has **5 hours** available to work this week. She will only be working in **neighborhood A** and only on **Saturday**. Her car gets **28 miles per gallon** and she expects to drive **60 miles**. The cost of fuel in her area is about **$2.95**.

Based on these values, Jane Doe is expected to make **$103.93**.

To do this you will need to plug in these values into our functions. Make sure your variable names match the parameters.

Function Name	Parameters	Return Type
`def amount_made_func():`	`base_pay, num_jobs_per_hr, hours, day_multiplier, neighborhood_multiplier`	int
`def deduction_func():`	`miles_traveled, fuel_efficiency, cost_of_fuel`	int
`def value_func():`	`amount_made, deduction`	int

Each of these functions will have a return type; in this case, every function should return an integer. Each of these parameters, `besides day_multiplier` and `neighborhood_multiplier`, should also be an integer. `day_multiplier` and `neighborhood_multiplier` will be run using the `checkList` function like previously mentioned.

Reflection: What happens if you change one variable? What combination of variables (of the ones the driver has control over) gives the highest expected income?

Part 3: Tying it Together: User Input

Allow the driver to enter their preferences and see their estimated weekly income.

Next, make your program interactive. Replace hardcoded variables with user input. This allows different drivers to use your calculator, that is, from the command line 🙂.

For every driver set variable, change these to take in user input using the input() function. Then use these variables as the parameters for the functions you've written in the previous sections. Pay attention to data types and have descriptive instructions for the user when typing input.

Here is an example of asking for the user's name, add this to your code as well.

```Python
print(input("What is your name?: "))
```

At the end, print a personalized message like the following:

```Python
"Jane Doe is expected to make: $103.93"
```

Congratulations! You created a program for delivery drivers to calculate their expected income. Deleting any “TODO” comments, and be sure to include your own comments and follow good naming conventions for variables.

Part 4: Ethical Considerations in Gig Work

Continue learning about gig work and responsible programming as it relates.

What are the Harms of Gig Work?

While gig work has its benefits, these digital platforms pose serious harms for workers.

Such concerns have positioned gig work as an area of research in human-computer interaction (HCI). Researchers in HCI and other areas point out concerns such as: low wages, job insecurity, lack of benefits, and physical hazards [3]. ***Reflection***: Do some of your own research, what are other harms gig workers face? Specifically, think about what problems might arise when software, not people, control a work environment.

Responsibility

Gig work falls into the idea that “software can only solve some problems, and in many cases, creates new ones” [4]. ***Reflection***: Do you associate computing with solving or causing problems? What responsibilities do programmers have when writing code that affects people’s livelihoods?

Reflection: Continue with your own research, what are proposed solutions to the harms of gig work? Your assignment is based on the “Work Planner” in the paper, “Stakeholder-Centered AI Design: Co-Designing Worker Tools with Gig Workers through Data Probes” by Zhang, Angie, et al.

“The future of work is already here. Workers around the world are increasingly hired, compensated, disciplined, and fired by algorithms that can be opaque, error-prone, and discriminatory; their faces, office badge swipes, email exchanges, browsing histories, keystrokes, driving patterns, and rest times are scanned to monitor performance and productivity; even activities outside working hours, such as health and fitness habits, social media usage, and attempts to organize are susceptible to employer surveillance and scrutiny.”

Sources

[1] https://guides.loc.gov/gig-economy/introduction
[2] https://www.pewresearch.org/internet/2021/12/08/the-state-of-gig-work-in-2021/
[3] https://dl.acm.org/doi/pdf/10.1145/3596671.3598576
[4] https://dl.acm.org/doi/pdf/10.1145/3424000
[5] https://dl.acm.org/doi/10.1145/3544548.3581354

Grading Rubric

Criteria	Proficient	Exemplary
Logic	• Calculations within functions are done correctly. • The functions work with little to no errors (i.e. errors are not noticeable or don't impede usage). • All the types are as specified in the instructions • Passes ≥10 of the Pytests.	• Meets all the Proficient requirements. • The program works with no errors. • Passes all of the Pytests.
Style	• Includes in-line comments. • Variable names are descriptive. • Unused code is deleted instead of commented out • A Pylint score between 7.0 - 8.0.	• Meets all the Proficient requirements. • Comments are descriptive, help improve readability, but aren't redundant. • Pylint score of ≥ 8.0
Reflection	• Answers all reflection questions.	• Meets all the Proficient requirements. • Answers to reflection questions are thoughtful. This includes using one or more of the following: ○ Real-world examples ○ Personal connections ○ Citing an article they read