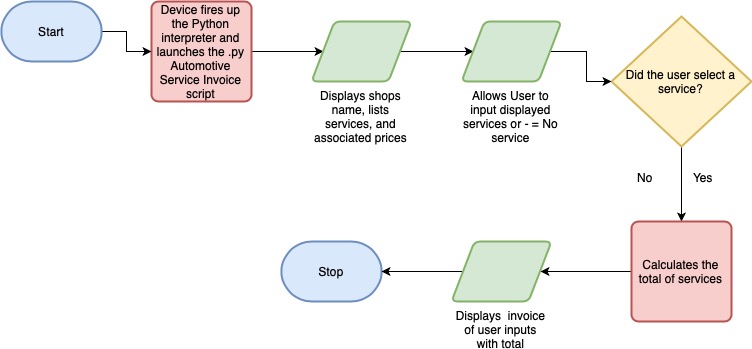
**Flowchart for Automobile Service Invoice:**



**Automobile Service Invoice Code:**

# Creates service option variables with associated prices

# **Best Practice 1**: annotated my code in sequences or chunks

print('Davy\'s auto shop services')

#**Best Practice 2:** ignored at this point, by not keeping variables consistently lower case

#**Problem-Solving 1:** Practiced with lists and dicts to test output, ended up scraping my original concept and using plan old variables with values best on program needed results.

Oil\_Change = 35

Tire\_Rotation = 19

Car\_Wash = 7

Car\_Wax = 12

no\_service = '-'

# Displays output of services and associated costs

# **Problem-Solving 2:** Referenced past Zybooks course material to accomplish the below substitution methods in order to eliminate spaces required by the program

print('Oil change -- $%d' % (Oil\_Change))

print('Tire rotation -- $%d' % (Tire\_Rotation))

print('Car wash -- $%d' % (Car\_Wash))

print('Car wax -- $%d' % (Car\_Wax))

print(")

# Allows User to input selected services

**# Best Practice 3:** Instead of focusing on using single quotes where statements where simple and double when complex I tried to be consistent and definitely need to work on this concept.

service\_1 = str(input("Select first service: "))

print('\n')

service\_2 = str(input("Select second service: \n"))

print('')

print('')

# Displays an Invoice to the Customer with previously selected services, associated costs, and re-assigns variables with ints

# **Best Practices 4:** Focused on 4 column indentation while writing if statements

# **Problem-Solving 3:** These if statements took a major transition using different concepts from the previous code written and trial and error.

# **Best Practices 5:** This block of code taught me to stay one step ahead, because I originally wrote it a different way, but when it came to the next step, I couldn’t get the total to simply add. print("Davy\'s auto shop invoice\n")

if service\_1 == 'Oil change':

print("Service 1:", "%s, $%d" % (service\_1, Oil\_Change))

service\_1 = 35

if service\_1 == 'Tire rotation':

print("Service 1:", "%s, $%d" % (service\_1, Tire\_Rotation))

service\_1 = 19

if service\_1 == 'Car wash':

print("Service 1:", "%s, $%d" % (service\_1, Car\_Wash))

service\_1 = 7

if service\_1 == 'Car wax':

print("Service 1:", "%s, $%d" % (service\_1, Car\_Wax))

service\_1 = 12

if service\_1 == '-':

print("Service 1: No service")

service\_1 = 0

if service\_2 == 'Oil change':

print("Service 2:", "%s, $%d\n" % (service\_2, Oil\_Change))

service\_2 = 35

if service\_2 == 'Tire rotation':

print("Service 2:", "%s, $%d\n" % (service\_2, Tire\_Rotation))

service\_2 = 19

if service\_2 == 'Car wash':

print("Service 2:", "%s, $%d\n" % (service\_2, Car\_Wash))

service\_2 = 7

if service\_2 == 'Car wax':

print("Service 2:", "%s, $%d\n" % (service\_2, Car\_Wax))

service\_2 = 12

if service\_2 == '-':

print("Service 2: No service\n")

service\_2 = 0

# Prints total based on user service selections

total\_services = service\_1 + service\_2

print('Total: $%d' % (total\_services))

**Programming Best Practice & Problem Solving:**

When writing this code, I kept referring back to the best practices this course has taught. Some of the items I focused on were comments and where to place them. I based my placement decision based on the best practice to code and correct in small chunks. It seemed rather intuitive to place the comments where the code begins a new process. I also focused on 4 column indentation instead of using tab while writing if statements. One item, I did not follow through with was keeping variable names lowercase throughout my code.

This program took a lot of trial and error to complete. I kept referencing past elements that were vague or fuzzy in my knowledge bank. I tried multiple approaches with items, along the programs sequences, that seemed incompatible in one way or another with the overall results. Dictionary versus a list, was a great example that I played with. While trying to figure out how to get the prices to print for the user to see without unwanted commas or colons. I ended up scrapping the concept and going with plan old variables with values assigned. I ultimately learned, that code evolves, and it is best to keep the next sequence in your mind.