

Session 4 Handout

1. Using while loops

The following code assumes that you have a correct `orderQuantity` function from last session in a `session3.py` file in the current directory.

```
[1]: from session3 import orderQuantity
    while True:
        userInput=input('Enter inventory (or done): ')
        if userInput=='done':
            break
        elif userInput=='skip':
            continue
        inventory=int(userInput)
        print ('Order',orderQuantity(inventory),'units.')
```

```
Enter inventory (or done): skip
Enter inventory (or done): 30
Order 70 units.
Enter inventory (or done): 25
Order 75 units.
Enter inventory (or done): done
```

Alternative implementation without using `break` or `continue`.

```
[ ]: userInput=input('Enter inventory (or done): ')
    while userInput!='done':
        if userInput!='skip':
            inventory=int(userInput)
            print ('Order',orderQuantity(inventory),'units.')
        userInput=input('Enter inventory (or done): ')
```

Q1: Write a program to repeatedly ask the user to input the number of hours worked, and display the total pay, assuming that the rate for first 40 hours is 10/hour, and the rate for additional hours is 15/hour. The program should terminate whenever the user inputs `done`.

[2]:

```
Enter hours worked (or done): 38
Pay is 380.0
Enter hours worked (or done): 42
Pay is 430.0
Enter hours worked (or done): done
```

(optional) Q2: Rewrite the code in Q1 but do not use `break`.

The following function uses `try` and `except` (see PY4E Chapter 3) for checking whether a certain value is convertible to a float.

```
[3]: def isNumber(x):
    try:
```

```

        float(x)
        return True
    except:
        return False

print(isNumber(3))
print(isNumber('3'))
print(isNumber('three'))

```

True
True
False

Q3: Modify the first example of this handout so that if the user does not input done nor an integer, then the program prints `Invalid input.` and asks for another input. (Hint: first write an `isInteger(x)` function by modifying the above, then use an `if` statement to decide whether to convert the input to an integer, or display `Invalid input.`)

[4]:

```

Enter inventory (or done): 2.5
Invalid input.
Enter inventory (or done): 30
Order 70 units.
Enter inventory (or done): thirty
Invalid input.
Enter inventory (or done): done

```

2. Using for loops

```

[5]: for i in [0,3,5,2]:
    print(i,end=' ')

```

0 3 5 2

```

[6]: for i in range(5):
    print(i,end=' ')

```

0 1 2 3 4

Q4: Modify the first example of the handout to use a `for` loop instead of a `while` loop, and limit the number of iterations to at most 5.