Problem:

Making Change. You are given n types of coin denominations of values v(1) < v(2) < ... < v(n) (all integers). Assume v(1) = 1, so you can always make change for any amount of money C. Give an algorithm which makes change for an amount of money C with as few coins as possible. [on problem set 4]

```
class Change():
  def __init__(self):
    print("Change calculator\n")
    while True:
      try:
        payment = float(input("Amount to pay: "))
        bill = float(input("Amount given: "))
        print("Change:", Change.makingChange(payment, bill))
        x = int(input("1. Quit\n2. Continue\n-"))
        if x == 1:
          print("Thank you for using this application!")
          break
        elif x == 2:
          continue

      except ValueError:
        print("Please input a number!")

  def makingChange(a, b):
    if a > b:
      print("You lack ", a-b, " in payment, please pay again!")
      return
    else:
      c = b - a

    return c

  def coinValues(x, y):
    z = x - y
    coin = (1, 5, 10, 20)
    for amount in z:
      return coin


Change()
```

```
Change calculator

Amount to pay: 5
Amount given: 1
You lack  4.0  in payment, please pay again!
Change: None
1. Quit
2. Continue
-2
Amount to pay: 10
Amount given: 50
Change: 40.0
1. Quit
2. Continue
-2
Amount to pay: 50
Amount given: s
Please input a number!
Amount to pay: 1
Amount given: 2
Change: 1.0
1. Quit
2. Continue
-1
Thank you for using this application!
<__main__.Change at 0x7b0e1b1f6950>
```

In making the code, I utilized bottom-up approach by making the interface for the users first so that it would be easier for me to understand what processes I would need for these interfaces.