CS 160
CS Orientation
More Programming Structure
Chap. 6 and 7
Four Ways to Design a Set of Conditions

Conditions:

- \( \text{Bonus} = 10 \) when \( \text{Pay} \leq 1000 \)
- \( \text{Bonus} = 50 \) when \( 1000 < \text{Pay} \leq 2000 \)
- \( \text{Bonus} = 100 \) when \( \text{Pay} > 2000 \)
1\textsuperscript{st} solution...

```python
pay = float(input("Enter pay: "))
if pay <= 1000:
    print("You get $10 bonus")
elif pay <= 2000:
    print("You get $50")
else:
    print("You get $100")
```
Four Ways to Design a Set of Conditions

Conditions:
- \( \text{Bonus} = 10 \) when \( \text{Pay} \leq 1000 \)
- \( \text{Bonus} = 50 \) when \( 1000 < \text{Pay} \leq 2000 \)
- \( \text{Bonus} = 100 \) when \( \text{Pay} > 2000 \)
2nd solution...

```
1 pay = float(input("Please enter the payment amount: "))
2
3 if(pay > 2000):
4   print("You get a $100 bonus!")
5 elif(pay > 1000):
6   print("You get a $50 bonus!")
7 else:
8   print("You get a $10 bonus!")
```
Four Ways to Design a Set of Conditions

Conditions:

- \( \text{Bonus} = 10 \) when \( \text{Pay} \leq 1000 \)
- \( \text{Bonus} = 50 \) when \( 1000 < \text{Pay} \leq 2000 \)
- \( \text{Bonus} = 100 \) when \( \text{Pay} > 2000 \)
3rd solution...

```python
pay = input("Please enter the payment amount: ")
pay = int(pay)

if (pay > 1000):
    if (pay > 2000):
        Bonus = 100
    else:
        Bonus = 50
else:
    Bonus = 10

print(Bonus)
```
Four Ways to Design a Set of Conditions

Conditions:

- **Bonus** = 10 when **Pay** <= 1000
- **Bonus** = 50 when 1000 < **Pay** <= 2000
- **Bonus** = 100 when **Pay** > 2000
4th solution...

```python
1 pay = input("Please enter the payment amount: ")
2 pay=int(pay);
3 if (pay <=2000):
4   if(pay<=1000):
5     Bonus=10;
6   else:
7     Bonus=50;
8 else:
9   Bonus=100;
10
11 print ("Hello, thy Bonus beith: "+str(Bonus))
```
### Fantastic Floral Company

#### Problem Analysis Chart

<table>
<thead>
<tr>
<th>Given Data</th>
<th>Required Results</th>
</tr>
</thead>
<tbody>
<tr>
<td>Wholesale buyers:</td>
<td></td>
</tr>
<tr>
<td>• Must have resale number.</td>
<td></td>
</tr>
<tr>
<td>• Amount of purchase discount</td>
<td></td>
</tr>
<tr>
<td>&lt; 100</td>
<td>2%</td>
</tr>
<tr>
<td>&gt;= 100 AND &lt; 500</td>
<td>5%</td>
</tr>
<tr>
<td>&gt;= 500</td>
<td>10%</td>
</tr>
<tr>
<td>Retail buyers:</td>
<td></td>
</tr>
<tr>
<td>• No discount</td>
<td></td>
</tr>
<tr>
<td>Amount of purchase (AP)</td>
<td></td>
</tr>
<tr>
<td>6% Tax</td>
<td></td>
</tr>
<tr>
<td>Amount owed</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Processing Required</th>
<th>Solution Alternatives</th>
</tr>
</thead>
<tbody>
<tr>
<td>Retail = 1.06 * AP</td>
<td></td>
</tr>
<tr>
<td>Wholesale = (1 − D) * AP</td>
<td></td>
</tr>
<tr>
<td>* Use positive logic for discount</td>
<td></td>
</tr>
<tr>
<td>* Use negative logic for discount</td>
<td></td>
</tr>
<tr>
<td>* Use straight-through logic for discount</td>
<td></td>
</tr>
</tbody>
</table>

*This solution was chosen because it results in the fewest number of tests and it is easily read.*
A different solution...

```python
1 type = input("Are you wholesale or retail (w or r)?")
2 AP = float(input("Enter your amount of purchase:"))
3 tax = .06
4 owed = 0
5 if(type == "w"):
6     resale_num = input("Enter your resale number:")
7     if(AP < 100):
8         discount = .02
9     elif(AP >= 100 and AP < 500):
10        discount = .05
11     else:
12        discount = .10
13     owed = (1-discount) * AP
14 elif(type == "r"):
15     owed = (1+tax) * AP
16 else:
17     print("You entered an invalid input!!!")
18 print("You owe: " + str(owed))
```