Bob Lin, Math381, hw1

First question(maximize value):

My problem is to choose from 50 objects each with value $floor(100 + 50\cos(i))$ of ith object and to make these objects has maximum value I can get. However, for each object I can only choose them one time, and due to issue of storage, I cannot let the total object I take to excess the threshold that I can store which is 1000 kg and 1000 liters. For each object, its value is a constant which is $floor(100 + 50\cos(i))$, then I will consider the constraint of weight and volume, which need to less or equal to 1000, and for each ith object, the weight and volume can be calculated by $floor(100 + 50\cos(5i - 3))a_i \ where\ a_i\ is\ the\ binary\ number\ of\ if\ we\ want\ that\ object\ and$ $floor(100 + 50\cos(5i - 3))a_i\ seperately$. By taking these two constraint into consideration, I want to maximize the value I want which is $\sum_{i=1}^{50} floor(100 + 50\cos(i))a_i$ that satisfies the constraint described above, also the only variable is a binary which is if we want to take that object(1 is yes and 0 is no), after considering all of these, I can start to come up with mathematic formulation to properly set the max value we want and constraint, and coefficient is the relation between index and given property named value, weight and volume, for value I use $floor(100 + 50\cos(i))$ to get it. So this become a linear problem, because I transfer each cos and floor to a constant, then for each ai, its coefficient is just a constant, so I can definitely adapt Lp to solve it easily!

To get the value:

$$\sum_{i=1}^{50} floor(100 + 50\cos(i))a_i\ (with\ a_i = \{0,1\})$$

This expression is to get the total value with binary value for each ai, and the purpose is to maximize it after adding following two constraint:

With constraint:

Weight:

$$\sum_{i=1}^{50} floor(100 + 50\cos\ (5i - 3))a_i \leq 1000(with\ a_i = \{0,1\})$$

This expression is to limit the total weight to at most 1000 with binary value for each ai

Volume:

$$\sum_{i=1}^{50} floor(100 + 50\cos\ (3i + 11))a_i \leq 1000(with\ a_i = \{0,1\})$$

This expression is to limit the total volume to at most 1000 with binary value for each ai

So, to sum up, the LP I am solving is to maximize the linear function of the calculation of value I can get from part of 50 objects I choose which satisfy their total volume and weight must not excess 1000.

Then, with the same idea, I write the code to create same formulation as below:

```python
import math

output = "max: "
for i in range(1, 51):
    val = math.floor(100 + 50 * math.cos(i))
    output = output + "+" + str(val) + "*" + "a_" + str(i)
output = output + ";"
print(output)

output = ""
for i in range(1,51):
    weight = 100 + math.floor(50 * math.cos(5 * i - 3))
    output = output + "+" + str(weight) + "*" + "a_" + str(i)
output = output + "<=1000;"
print(output)

output = ""
for i in range(1,51):
    volume = 100 + math.floor(50 * math.cos(3 * i + 11))
    output = output + "+" + str(volume) + "*" + "a_" + str(i)
output = output + "<=1000;"
print(output)

output = "bin "
for i in range(1,50):
    output = output + "a_" + str(i) + ","
output = output + "a_50;"
print(output)
```

then this create lp solver input file which is below:

```
max:
+127*a_1+79*a_2+50*a_3+67*a_4+114*a_5+148*a_6+137*a_7+92*a_8+54*a_9+58*a_10+100*a_11+142*a_12+145*a_13+106*a_14+62*a_15+52*a_16+86*a_17+133*a_18+149*a_19+120*a_20+72*a_21+50*a_22+73*a_23+121*a_24+149*a_25+132*a_26+85*a_27+51*a_28+62*a_29+107*a_30+145*a_31+141*a_32+99*a_33+57*a_34+54*a_35+93*a_36+138*a_37+147*a_38+113*a_39+66*a_40+50*a_41+80*a_42+127*a_43+149*a_44+126*a_45+78*a_46+50*a_47+67*a_48+115*a_49+148*a_50;

+79*a_1+137*a_2+142*a_3+86*a_4+50*a_5+85*a_6+141*a_7+138*a_8+80*a_9+50*a_10+91*a_11+144*a_12+133*a_13+74*a_14+51*a_15+98*a_16+147*a_17+128*a_18+68*a_19+53*a_20+105*a_21+149*a_22+122*a_23+63*a_24+56*a_25+111*a_26+149*a_27+116*a_28+59*a_29+60*a_30+117*a_31+149*a_32+110*a_33+56*a_34+64*a_35+123*a_36+148*a_37+103*a_38+53*a_39+69*a_40+129*a_41+147*a_42+97*a_43+51*a_44+75*a_45+134*a_46+144*a_47+90*a_48+50*a_49+81*a_50<=1000;

+106*a_1+86*a_2+120*a_3+73*a_4+132*a_5+62*a_6+141*a_7+54*a_8+147*a_9+50*a_10+149*a_11+50*a_12+148*a_13+54*a_14+142*a_15+61*a_16+133*a_17+71*a_18+122*a_19+84*a_20+108*a_21+98*a_22+94*a_23+112*a_24+80*a_25+125*a_26+68*a_27+136*a_28+59*a_29+144*a_30+52*a_31+149*a_32+50*a_33+149*a_34+51*a_35+146*a_36+56*a_37+139*a_38+65*a_39+129*a_40+76*a_41+116*a_42+90*a_43+102*a_44+104*a_45+88*a_46+117*a_47+75*a_48+130*a_49+64*a_50<=1000;
```

```
bin
a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8,a_9,a_10,a_11,a_12,a_13,a_14,a_15,a_16,a_17,a_18,a_19,a_20,a_21,
a_22,a_23,a_24,a_25,a_26,a_27,a_28,a_29,a_30,a_31,a_32,a_33,a_34,a_35,a_36,a_37,a_38,a_39,a_40
,a_41,a_42,a_43,a_44,a_45,a_46,a_47,a_48,a_49,a_50;
```

then this come up with solution in Lp solver which is below:

```
Value of objective function: 1680.00000000


Actual values of the variables:

a_1                    1

a_2                    0

a_3                    0

a_4                    0

a_5                    0

a_6                    1

a_7                    0

a_8                    0

a_9                    0

a_10                    1

a_11                    0

a_12                    1

a_13                    0

a_14                    1

a_15                    0

a_16                    0

a_17                    0

a_18                    0

a_19                    1

a_20                    1

a_21                    0

a_22                    0
```

a_23	0
a_24	0
a_25	1
a_26	0
a_27	0
a_28	0
a_29	0
a_30	0
a_31	1
a_32	0
a_33	0
a_34	0
a_35	0
a_36	0
a_37	0
a_38	0
a_39	1
a_40	0
a_41	0
a_42	0
a_43	0
a_44	1
a_45	1
a_46	0
a_47	0
a_48	0
a_49	0
a_50	1

which means we need is 1,6,10,12,14,19,20,24,25,31,39,44,45,50

The variation2 is to let me have unlimited number of a_1 to a_50, with same constraint of weight and volume and still looking for the largest value it can output, which is can let all a_1 to 1_50 be non negative integer, and just switch the bin to integer to let it can be as large as possible for a given variable, so input file become：

```
max:
+127*a_1+79*a_2+50*a_3+67*a_4+114*a_5+148*a_6+137*a_7+92*a_8+54*a_9+58*a_10+100*a_11+
142*a_12+145*a_13+106*a_14+62*a_15+52*a_16+86*a_17+133*a_18+149*a_19+120*a_20+72*a_2
1+50*a_22+73*a_23+121*a_24+149*a_25+132*a_26+85*a_27+51*a_28+62*a_29+107*a_30+145*a_3
1+141*a_32+99*a_33+57*a_34+54*a_35+93*a_36+138*a_37+147*a_38+113*a_39+66*a_40+50*a_41
+80*a_42+127*a_43+149*a_44+126*a_45+78*a_46+50*a_47+67*a_48+115*a_49+148*a_50;

+79*a_1+137*a_2+142*a_3+86*a_4+50*a_5+85*a_6+141*a_7+138*a_8+80*a_9+50*a_10+91*a_11+1
44*a_12+133*a_13+74*a_14+51*a_15+98*a_16+147*a_17+128*a_18+68*a_19+53*a_20+105*a_21+1
49*a_22+122*a_23+63*a_24+56*a_25+111*a_26+149*a_27+116*a_28+59*a_29+60*a_30+117*a_31+
149*a_32+110*a_33+56*a_34+64*a_35+123*a_36+148*a_37+103*a_38+53*a_39+69*a_40+129*a_4
1+147*a_42+97*a_43+51*a_44+75*a_45+134*a_46+144*a_47+90*a_48+50*a_49+81*a_50<=1000;

+106*a_1+86*a_2+120*a_3+73*a_4+132*a_5+62*a_6+141*a_7+54*a_8+147*a_9+50*a_10+149*a_11
+50*a_12+148*a_13+54*a_14+142*a_15+61*a_16+133*a_17+71*a_18+122*a_19+84*a_20+108*a_21
+98*a_22+94*a_23+112*a_24+80*a_25+125*a_26+68*a_27+136*a_28+59*a_29+144*a_30+52*a_31+
149*a_32+50*a_33+149*a_34+51*a_35+146*a_36+56*a_37+139*a_38+65*a_39+129*a_40+76*a_41+
116*a_42+90*a_43+102*a_44+104*a_45+88*a_46+117*a_47+75*a_48+130*a_49+64*a_50<=1000;

Int

a_1,a_2,a_3,a_4,a_5,a_6,a_7,a_8,a_9,a_10,a_11,a_12,a_13,a_14,a_15,a_16,a_17,a_18,a_19,a_20,a_21,
a_22,a_23,a_24,a_25,a_26,a_27,a_28,a_29,a_30,a_31,a_32,a_33,a_34,a_35,a_36,a_37,a_38,a_39,a_40
,a_41,a_42,a_43,a_44,a_45,a_46,a_47,a_48,a_49,a_50;
```

In the variance3, this is I can have decimal count of a_1 to a_50, with still the same amount of weight and volume and still looking for the greatest value I can get, so I let all a_1 to 1_50 can be any number non-negative, which just need to delete the last line because in lp solver the default range for variable is non-negative, then this become:

```
max:
+127*a_1+79*a_2+50*a_3+67*a_4+114*a_5+148*a_6+137*a_7+92*a_8+54*a_9+58*a_10+100*a_11+
142*a_12+145*a_13+106*a_14+62*a_15+52*a_16+86*a_17+133*a_18+149*a_19+120*a_20+72*a_2
1+50*a_22+73*a_23+121*a_24+149*a_25+132*a_26+85*a_27+51*a_28+62*a_29+107*a_30+145*a_3
1+141*a_32+99*a_33+57*a_34+54*a_35+93*a_36+138*a_37+147*a_38+113*a_39+66*a_40+50*a_41
+80*a_42+127*a_43+149*a_44+126*a_45+78*a_46+50*a_47+67*a_48+115*a_49+148*a_50;

+79*a_1+137*a_2+142*a_3+86*a_4+50*a_5+85*a_6+141*a_7+138*a_8+80*a_9+50*a_10+91*a_11+1
44*a_12+133*a_13+74*a_14+51*a_15+98*a_16+147*a_17+128*a_18+68*a_19+53*a_20+105*a_21+1
49*a_22+122*a_23+63*a_24+56*a_25+111*a_26+149*a_27+116*a_28+59*a_29+60*a_30+117*a_31+
149*a_32+110*a_33+56*a_34+64*a_35+123*a_36+148*a_37+103*a_38+53*a_39+69*a_40+129*a_4
1+147*a_42+97*a_43+51*a_44+75*a_45+134*a_46+144*a_47+90*a_48+50*a_49+81*a_50<=1000;
```

```
+106*a_1+86*a_2+120*a_3+73*a_4+132*a_5+62*a_6+141*a_7+54*a_8+147*a_9+50*a_10+149*a_11
+50*a_12+148*a_13+54*a_14+142*a_15+61*a_16+133*a_17+71*a_18+122*a_19+84*a_20+108*a_21
+98*a_22+94*a_23+112*a_24+80*a_25+125*a_26+68*a_27+136*a_28+59*a_29+144*a_30+52*a_31+
149*a_32+50*a_33+149*a_34+51*a_35+146*a_36+56*a_37+139*a_38+65*a_39+129*a_40+76*a_41+
116*a_42+90*a_43+102*a_44+104*a_45+88*a_46+117*a_47+75*a_48+130*a_49+64*a_50<=1000;
```