Information for objects that will be used in this project:

Suppose object i (where $1 \leq i \leq 100$) has the following value, weight and volume:

value = floor(50+25 cos(7i)) thousands of dollars

weight = floor(30+12 cos(6i+2)) kg

volume =floor(40+8 cos(5i+4)) liters

(a) ***Problem:***

Suppose every single object described above is unique. We have a container that has a volume capacity of 400L and a weight capacity of 500kg. Figure out what objects we should put into the container to maximize the total value.

Build the mathematical formula:

Let x_i ($i \in Z$, $i \in [1, 100]$) be the binary value such that $x_i = 0$ means that object i won't be put into the container and $x_i = 1$ means that object i will be put into the container.

This means our object function is $\sum_{i=1}^{100} x_i \cdot value_i$.

Since the volume and weight that could be in the container is limited, our constraints are $\sum_{i=1}^{100} x_i \cdot weight_i \leq 400$ and $\sum_{i=1}^{100} x_i \cdot volume_i \leq 500$.

Thus, my Lp is maximize $\sum_{i=1}^{100} x_i \cdot value_i$ subject to $\sum_{i=1}^{100} x_i \cdot weight_i \leq 400$ and $\sum_{i=1}^{100} x_i \cdot volume_i \leq 500$.

Code used to generate input file (JAVA):

```
1 import java.util.*;
2
3 public class math381hw1a {
4   public static void main(String[] args) {
5     // Print out the object function.
6     System.out.print("max: ");
7     for (int i = 1; i <= 100; i++) {
8       if (i < 100) {
9         System.out.print(Math.floor(50 + 25*Math.cos(7*i)) + "x_" + i + "+");
10      } else {
11        System.out.println(Math.floor(50 + 25*Math.cos(7*i)) + "x_" + i + ";");
```

```
12        }
13      }
14      // Print out the formula to calculate the sum of the weight of chosen quantities
15      // and the constraint for total weight.
16      System.out.print("weight = ");
17      for (int j = 1; j <= 100; j++) {
18        if (j < 100) {
19          System.out.print(Math.floor(30 + 12*Math.cos(6*j + 2)) + "x_" + j + "+");
20        } else {
21          System.out.println(Math.floor(30 + 12*Math.cos(6*j + 2)) + "x_" + j + ";");
22        }
23      }
24      System.out.println("weight <= 400;");
25      // Print out the formula to calculate the sum of the volume of chosen quantities
26      // and the constraint for total volume.
27      System.out.print("volume = ");
28      for (int k = 1; k <= 100; k++) {
29        if (k < 100) {
30          System.out.print(Math.floor(40 + 8*Math.cos(5*k + 4)) + "x_" + k + "+");
31        } else {
32          System.out.println(Math.floor(40 + 8*Math.cos(5*k + 4)) + "x_" + k + ";");
33        }
34      }
35      System.out.println("volume <= 500;");
36      // Make the command that this is an integer program and every object could be
37      // chosen once only.
38      System.out.print("bin ");
39      for (int l = 1; l <= 100; l++) {
40        if (l < 100){
41          System.out.print("x_" + l + ", ");
42        } else {
43          System.out.println("x_" + l + ";");
44        }
45      }
46    }
47  }
```

Input file:

max:
68.0x_1+53.0x_2+36.0x_3+25.0x_4+27.0x_5+40.0x_6+57.0x_7+71.0x_8+74.0x_9+65.0x_10+49.0x_11+32.0x_12+25.0x_13+29.0x_14+43.0x_15+61.0x_16+73.0x_17+73.0x_18+62.0x_19+45.0x_20+30.0x_21+25.0x_22+32.0x_23+48.0x_24+64.0x_25+74.0x_26+71.0x_27+58.0x_28+41.0x_29+27.0x_30+25.0x_31+35.0x_32+52.0x_33+68.0x_34+74.0x_35+69.0x_36+54.0x_37+37.0x_38+26.0x_39+26.0x_40+38.0x_41+56.0x_42+70.0x_43+74.0x_44+66.0x_45+50.0x_46+33.0x_47+25.0x_48+28.0x_49+42.0x_50+60.0x_51+72.0x_52+73.0x_53+63.0x_54+46.0x_55+30.0x_56+25.0x_57+31.0x_58+47.0x_59+64.0x_60+74.0x_61+72.0x_62+59.0x_63+42.0x_64+28.0x_65+25.0x_66+34.0x_67+51.0x_68+67.0x_69+74.0x_70+70.0x_71+55.0x_72+38.0x_73+26.0x_74+26.0x_75+38.0x_76+55.0x_77+70.0x_78+74.0x_79+67.0x_80+51.0x_81+34.0x_82+25.0x_83+28.0x_84+41.0x_85+59.0x_86+72.0x_87+74.0x_88+64.0x_89+47.0x_90+31.0x_91+25.0x_92+30.0x_93+45.0x_94+63.0

x_95+73.0x_96+72.0x_97+60.0x_98+43.0x_99+29.0x_100;
weight =
28.0x_1+31.0x_2+34.0x_3+37.0x_4+40.0x_5+41.0x_6+41.0x_7+41.0x_8+40.0x_9+38.0x_10+35.0x_11+32.0
x_12+28.0x_13+25.0x_14+22.0x_15+20.0x_16+18.0x_17+18.0x_18+18.0x_19+19.0x_20+21.0x_21+24.0x_2
2+27.0x_23+31.0x_24+34.0x_25+37.0x_26+39.0x_27+41.0x_28+41.0x_29+41.0x_30+40.0x_31+38.0x_32+3
5.0x_33+32.0x_34+29.0x_35+25.0x_36+22.0x_37+20.0x_38+18.0x_39+18.0x_40+18.0x_41+19.0x_42+21.0x
_43+23.0x_44+27.0x_45+30.0x_46+33.0x_47+36.0x_48+39.0x_49+41.0x_50+41.0x_51+41.0x_52+40.0x_53
+38.0x_54+36.0x_55+33.0x_56+29.0x_57+26.0x_58+23.0x_59+20.0x_60+19.0x_61+18.0x_62+18.0x_63+19.
0x_64+20.0x_65+23.0x_66+26.0x_67+29.0x_68+33.0x_69+36.0x_70+38.0x_71+40.0x_72+41.0x_73+41.0x_
74+41.0x_75+39.0x_76+36.0x_77+33.0x_78+30.0x_79+27.0x_80+24.0x_81+21.0x_82+19.0x_83+18.0x_84+
18.0x_85+18.0x_86+20.0x_87+22.0x_88+25.0x_89+29.0x_90+32.0x_91+35.0x_92+38.0x_93+40.0x_94+41.0
x_95+41.0x_96+41.0x_97+39.0x_98+37.0x_99+34.0x_100;
weight <= 400;
volume =
32.0x_1+41.0x_2+47.0x_3+43.0x_4+34.0x_5+33.0x_6+42.0x_7+47.0x_8+42.0x_9+33.0x_10+33.0x_11+43.0
x_12+47.0x_13+41.0x_14+32.0x_15+34.0x_16+44.0x_17+47.0x_18+40.0x_19+32.0x_20+35.0x_21+44.0x_2
2+47.0x_23+39.0x_24+32.0x_25+36.0x_26+45.0x_27+46.0x_28+38.0x_29+32.0x_30+37.0x_31+46.0x_32+4
6.0x_33+37.0x_34+32.0x_35+38.0x_36+47.0x_37+45.0x_38+36.0x_39+32.0x_40+39.0x_41+47.0x_42+44.0x
_43+35.0x_44+32.0x_45+40.0x_46+47.0x_47+44.0x_48+34.0x_49+32.0x_50+41.0x_51+47.0x_52+43.0x_53
+33.0x_54+33.0x_55+42.0x_56+47.0x_57+42.0x_58+33.0x_59+34.0x_60+43.0x_61+47.0x_62+41.0x_63+32.
0x_64+34.0x_65+44.0x_66+47.0x_67+39.0x_68+32.0x_69+35.0x_70+45.0x_71+47.0x_72+38.0x_73+32.0x_
74+36.0x_75+45.0x_76+46.0x_77+37.0x_78+32.0x_79+37.0x_80+46.0x_81+46.0x_82+36.0x_83+32.0x_84+
38.0x_85+47.0x_86+45.0x_87+35.0x_88+32.0x_89+39.0x_90+47.0x_91+44.0x_92+35.0x_93+32.0x_94+40.0
x_95+47.0x_96+43.0x_97+34.0x_98+33.0x_99+41.0x_100;
volume <= 500;
(ensure all variables are binary)
bin x_1, x_2, …, x_99, x_100;

Output file by using lp_solve to generate the input file:

Value of objective function: 1000.00000000

Actual values of the variables:

x_1	1
x_9	1
x_17	1
x_26	1
x_35	1
x_44	1
x_45	1
x_61	1
x_69	1
x_70	1
x_78	1
x_79	1
x_88	1
x_89	1

All other variables are zero.

weight	400

volume 499

Concluding solution:

With the information from the output file of (a) listed below, we could maximize the total value as 1000 thousands dollars in the container by putting object 1, 9, 17, 26, 35, 44, 45, 61, 69, 70, 78, 79, 88, 89 into the container.

Checking binding information:

Only the constraint of weight is binding. By checking the output file of (a) listed below, the container finally has a total weight of 400kg and a total volume of 499L.

400kg is the constraint of total weight for the container, but the constraint of total volume for the container is 500L, which is greater than the actual total volume of the container.

(b) ***Problem:***

Suppose there is a lower bound on the number of objects we put into the container. Find out how different lower bounds will change the total value in the container.

Build the mathematical formula:

Our object function is the same as the object function in (a). We have another constraint: $\sum_{i=1}^{100} x_i \geq k$ where k is the lower bound of number of objects in the container.

Thus, my Lp is maximize $\sum_{i=1}^{100} x_i \cdot value_i$ subject to $\sum_{i=1}^{100} x_i \cdot weight_i \leq 400$, $\sum_{i=1}^{100} x_i \cdot volume_i \leq 500$, and $\sum_{i=1}^{100} x_i \geq k$.

Code used to generate input file (JAVA. Add the following after line 35 in (a)'s code used to generate input file, and when the lower bound is 16, just change the number in the following code from 15 to 16):

```
37    // In this case, the lower bound is 15.
38    System.out.println("total = ");
39    for (int m = 1; m <= 100; m ++) {
40      if (m < 100) {
41        System.out.print("x_" + m + "+");
```

```
42        } else {
43          System.out.println("x_" + m +";");
44        }
45      }
46      System.out.println("total >= 15;");
```

Input file (The following input will be added after the line "volume ≤ 500" in (a)'s input file when the lower bound if 15, and when the lower bound is 16, the last line in the following will be changed to "total ≥ 16"):

```
total =
x_1+x_2+x_3+x_4+x_5+x_6+x_7+x_8+x_9+x_10+x_11+x_12+x_13+x_14+x_15+x_16+x_17+x_18+x_19+x
_20+x_21+x_22+x_23+x_24+x_25+x_26+x_27+x_28+x_29+x_30+x_31+x_32+x_33+x_34+x_35+x_36+x_3
7+x_38+x_39+x_40+x_41+x_42+x_43+x_44+x_45+x_46+x_47+x_48+x_49+x_50+x_51+x_52+x_53+x_54+
x_55+x_56+x_57+x_58+x_59+x_60+x_61+x_62+x_63+x_64+x_65+x_66+x_67+x_68+x_69+x_70+x_71+x_
72+x_73+x_74+x_75+x_76+x_77+x_78+x_79+x_80+x_81+x_82+x_83+x_84+x_85+x_86+x_87+x_88+x_89
+x_90+x_91+x_92+x_93+x_94+x_95+x_96+x_97+x_98+x_99+x_100;
total >= 15;
```

(The output file when the lower bound ≤ 14 is exactly the output file in (a))

Output file by using lp_solve to generate the input file when lower bound is 15:

```
Value of objective function: 985.00000000

Actual values of the variables:
x_1                           1
x_16                          1
x_20                          1
x_25                          1
x_35                          1
x_36                          1
x_44                          1
x_45                          1
x_59                          1
x_60                          1
x_69                          1
x_70                          1
x_79                          1
x_88                          1
x_89                          1
All other variables are zero.
weight                      394
volume                      500
total                        15
```

Output file by using lp_solve to generate the input file when lower bound is 16:

This problem is infeasible

Discussing three different situations of lower bounds:

Since there are 14 objects in the container when there is no constraint for lower bound in (a), the value of the object function is the same (1000) when $0 \leq$ lower bound ≤ 14.

When the lower bound = 15, with the information form the output file of (b1) listed below, we could maximize the total value as 985 thousands dollars in the container by putting object 1, 16, 20, 25, 35, 36, 44, 45, 59, 60, 69, 70, 79, 88, 89 into the container when the lower bound equals to 15.

When the lower bound is greater than 15, this model will be infeasible.

Analyze:

Since $\cos(x) \geq -1$ for all real numbers x, an object has a minimum weight of 30 + 12*-1 = 18.

An object also has a minimum volume of 40 + 8*-1 = 32.

Regardless of volume, the maximum number of objects that can be put in the container is floor(400/18) = 22.

Regardless of weight, the maximum number of objects that can be put in the container is floor(500/32) = 15.

This means by the combination of weight and volume, there must be no more than 15 objects in the container, which is exactly what we get by using lp_solve.

Thus, all positive integers that are no bigger than 15 could be the lower bound of on the number of objects that we put into the container.

Objects 9, 17, 26, 61 are pulled out and objects 16, 20, 25, 36, 60 are pulled in when we change from the LP problem with lower bound no larger than 14 to the LP problem with lower bound equals 15. See the table below for comparison (green represents

objects that are pulled out and red represents objects that are put in).

By comparison, the objects that are pulled out have a relatively higher value, weight, and volume than the objects that are put in.

Object	9	16	17	20	25	26	36	60	61
Value	74	61	73	45	64	74	69	64	74
Weight	40	20	18	19	34	37	25	20	19
Volume	42	34	44	32	32	36	38	34	43

(c) ***Problem:***

Suppose every object is not unique. Figure out what objects should we put into the container to maximize the total value.

Build the mathematical formula:

The object function and constraints are exactly the same as a's, and x_i could be any non-negative integer in this case.

Code used to generate input file (JAVA, delete the added code from (b), then delete every line of code include and after line 36, and finally add the following code):

```
36      // Make the command that this is an integer program and every object could be
37      // chosen for unlimited times.
38      System.out.print("int ");
39      for (int l = 1; l <= 100; l++) {
40        if (l < 100){
41          System.out.print("x_" + l + ", ");
42        } else {
43          System.out.println("x_" + l + ";");
44        }
45      }
46    }
47 }
```

Input file (same as what in (a) except the last two line changes to the following):

(ensure all variables are integers)

int x_1, x_2, …, x_99, x_100;

Output file by using lp_solve to generate the input file:

Value of objective function: 1110.00000000

Actual values of the variables:
x_35 10
x_88 5
All other variables are zero.
weight 400
volume 495

Concluding solution:

With the information form the output file of (c) listed below, we could maximize the total value as 1110 thousands dollars in the container by putting 10 x_35 and 5 x_88 into the container.

Compare solution in (c) with (a):

We know by (b) that an object has a minimum weight of 18 and a minimum volume of 32.

Since $\cos(y) \leq 1$ for all real numbers y, an object has a maximum value of 50 + 25*1 = 75.

By comparing x_{35} and x_{88} with the objects in the concluding solution part in (a), I find out that both object 35 and 88 have a relatively larger value as a single object with relatively smaller volume and lighter weight as a single object. Check the table below.

Object	35	88
Value	74	74
Weight	29	22
Volume	32	35