**Problems to solve**

There are two options: Solving the problem with GAMS, or solving it by some meta-heuristic method (on some programming language). Each problem can be chosen by at most one student (except if one of them solves it by GAMS another by some heu method. Two students also can choose the same problem if they solve it by different heu methods like Genetic Algorithm and Tabu Search).

**1.** (based on Winston, page 420, Exercise 4.)

A company produces computers in 8 cities, A1,…,A8.
The amount that they can produce per year (at most) is as follows: 280, 400, 3000, 240, 1200, 430, 2300, 650.
The cost of production is as follows (per unit): 250, 650, 340, 970, 650, 340, 450, 650.
They transport the computers to 9 cities, C1,…,C9.
The demand is as follows: 230, 130, 250, 345, 650, 350, 320, 540, 1200.

There is also some initial cost if production is made, it is 2000 dollar at any factory.
The transportation goes through the next cities: B1,…,B5.
The transportation costs are given as follows (per unit):

| | B1 | B2 | B3 | B4 | B5 |
|---|---|---|---|---|---|
| A1 | 8 | 22 | 28 | 13 | 44 |
| A2 | 10 | 25 | 17 | 14 | 45 |
| A3 | 14 | 40 | 50 | 52 | 60 |
| A4 | 15 | 43 | 34 | 23 | 12 |
| A5 | 17 | 34 | 45 | 25 | 25 |
| A6 | 17 | 52 | 46 | 27 | 34 |
| A7 | 12 | 56 | 48 | 21 | 21 |
| A8 | 13 | 72 | 12 | 34 | 44 |

and

| | C1 | C2 | C3 | C4 | C5 | C6 | C7 | C8 | C9 |
|---|---|---|---|---|---|---|---|---|---|
| B1 | 12 | 15 | 35 | 16 | 23 | 23 | 43 | 26 | 23 |
| B2 | 33 | 8 | 24 | 15 | 24 | 25 | 44 | 25 | 25 |
| B3 | 34 | 14 | 31 | 12 | 11 | 26 | 35 | 24 | 62 |
| B4 | 23 | 13 | 32 | 15 | 12 | 27 | 60 | 22 | 43 |
| B5 | 25 | 28 | 42 | 15 | 15 | 28 | 18 | 21 | 23 |

It also holds that in the intermediate places (B1,…,B5) at most the following units can be transported: 1300, 1200, 750, 790, 2300.

Let us determine the optimal transportation plan ( how much should be transported from what place to what place through what cities) so that the total cost is minimum!

**2.** (based on Winston, page 441/6)
A factory produces some type of product, at 3 places (A and B and C), during 6 months.

The production cost per unit is as follows:

| | Month 1 | Month 2 | Month 3 | Month 4 | Month 5 | Month 6 |
|---|---|---|---|---|---|---|
| A | 33 | 22 | 28 | 95 | 48 | 15 |
| B | 35 | 18 | 32 | 84 | 55 | 23 |
| C | 43 | 22 | 25 | 65 | 45 | 35 |

If production is made at some place, an initial cost 1000 $ incurs.

The capacity (how many can be produced during the month) is as follows:

| | Month 1 | Month 2 | Month 3 | Month 4 | Month 5 | Month 6 |
|---|---|---|---|---|---|---|
| A | 6 | 7 | 9 | 7 | 2 | 4 |
| B | 5 | 7 | 4 | 8 | 9 | 10 |
| C | 4 | 2 | 5 | 9 | 4 | 5 |

Just when the product is made, it is transported to the only one buyer.

The shipping cost (per unit) is as follows:

| | Month 1 | Month 2 | Month 3 | Month 4 | Month 5 | Month 6 |
|---|---|---|---|---|---|---|
| A | 61 | 65 | 54 | 45 | 47 | 43 |
| B | 55 | 71 | 48 | 45 | 42 | 40 |
| C | 45 | 46 | 50 | 61 | 55 | 51 |

A transported product can be stored at the destination point. The storing cost is 13 per unit per month, and at most 12 units can be stored at any month.

The demand is as follows: 10, 17, 15, 4, 22, 23.

Determine the optimal production/transport policy (for which the total cost is as small as possible).

**3.** A company purchases, stores and sells, two kinds of products (Product1 and Product2), during a 6 month period. The data:

**Product1**

| | Month 1 | Month 2 | Month 3 | Month 4 | Month 5 | Month 6 |
|---|---|---|---|---|---|---|
| Buying price: | 3 | 5 | 8 | 12 | 14 | 23 |
| Storing cost: | 1 | 1 | 2 | 1 | 3 | 2 |
| Demand: | 30 | 45 | 18 | 25 | 33 | 42 |

**Product2**

| | Month 1 | Month 2 | Month 3 | Month 4 | Month 5 | Month 6 |
|---|---|---|---|---|---|---|
| Buying price: | 30 | 51 | 82 | 95 | 15 | 43 |
| Storing cost: | 5 | 7 | 4 | 6 | 7 | 9 |
| Demand: | 18 | 40 | 53 | 42 | 15 | 9 |

The products are stored in a common warehouse. Here, however, there is limited storage capacity. Products that are bought and sold immediately will not be in stock, so storage capacity only applies to other products that still exist on the 3rd day of the month. (First day: the day of purchasing, second day: the day of selling, from 3-th day to end of month: storing.)

| | Month 1 | Month 2 | Month 3 | Month 4 | Month 5 | Month 6 |
|---|---|---|---|---|---|---|
| **Storage capacity** | 20 | 20 | 30 | 30 | 20 | 20 |

For the sake of simplicity, we assume the following:

- the initial quantity stored is 0 for both products and the warehouses must be completely emptied by the end of the 6-month period
- the products are purchased on the first day of each month.
- All demands must be fully met, the demands will be met (by delivery) on the second day of the month.
- As much of the product remains (it is still in the storehouse on the third day of the month), it must be stored. So what is left on the third day of the month is stored during the month and this stored quantity can be used to meet the demand for the following month (or even later months).

A, Determine the optimal purchase / storage plan. So how much of which product to buy in which month (it already follows how much of the products should be stored). All costs (purchase price + storage costs) should be minimized.

B, To what extent do we get another solution if the stored goods cannot be stored for more than 2 months? So which goods we buy at the beginning of the first month can be stored in the first month and the second month, but must be sold no later than at the beginning of the third month. (In the warehouse, we distinguish in which month the purchased product was purchased).

**4.** (based on Winston 975/2)

Suppose we need a car for the next six years (we buy and maintain a car). Let’s say we just have a new car right now and the first year is just starting. The price of a new car is $ 10,000.

The annual costs and the selling price of the used car are given in the following table:

| Age of the car (years) | Selling price | Maintenance costs |
|---|---|---|
| 1 | $7000 | $300 |
| 2 | $6000 | $500 |
| 3 | $4000 | $800 |
| 4 | $3000 | $1200 |
| 5 | $2000 | $1600 |
| 6 | $1000 | $2200 |

A, Determine an optimal plan for how long to keep the car and buy a new car so that we want to minimize our total cost for 6 years. At the end of the 6 years period we keep our car (no matter how old is it).

B, let’s say we also need a combine (we just bought a new combine for $ 20,000).

The data for the combine are given in the following table:

| Age of the combine (in years) | Selling price | Maintenance costs |
|---|---|---|
| 1 | $15,000 | $3000 |
| 2 | $14,000 | $5000 |
| 3 | $12,000 | $9000 |
| 4 | $8000 | $12,000 |
| 5 | $7000 | $16,000 |
| 6 | $5000 | $19,000 |

Suppose we can't buy a combine and a car at the same time (in the same year).
Let us determine an optimal plan (when to buy a car or combine for how long to keep them) for the next six years (at the end we keep both our car and combine no matter how old are they).

**5. Bin packing**

Given the following n=120 items (Falkenauer t120_00 set, see, Unibo homepage)

497, 497, 495, 485, 480, 478, 474, 473, 472, 470, 466, 450, 446, 445, 445, 444, 439, 434, 430, 420, 419, 414, 412, 410, 407, 405, 400, 397, 395, 376, 372, 370, 366, 366, 366, 366, 366, 363, 363, 362, 361, 357, 357, 356, 356, 355, 352, 351, 350, 350, 350, 347, 336, 333, 329, 325, 320, 315, 314, 313, 307, 303, 302, 301, 299, 298, 298, 298, 295, 294, 292, 290, 288, 287, 283, 282, 282, 276, 275, 275, 274, 273, 273, 272, 272, 271, 271, 269, 269, 268, 267, 267, 266, 263, 263, 262, 262, 261, 260, 259, 259, 259, 258, 256, 255, 254, 254, 254, 253, 253, 253, 253, 252, 252, 252, 252, 251, 251, 250, 250.

The bin size is 1000.
Determine the optimal solution (how to pack the items into minimum number of C=1000 capacity bins). Give a „fast” solution, if possible.

**6. Class constrained bin packing.**

Given the following n=120 items (Falkenauer t120_00 set, see, Unibo homepage)

497, 497, 495, 485, 480, 478, 474, 473, 472, 470, 466, 450, 446, 445, 445, 444, 439, 434, 430, 420, 419, 414, 412, 410, 407, 405, 400, 397, 395, 376, 372, 370, 366, 366, 366, 366, 366, 363, 363, 362, 361, 357, 357, 356, 356, 355, 352, 351, 350, 350, 350, 347, 336, 333, 329, 325, 320, 315, 314, 313, 307, 303, 302, 301, 299, 298, 298, 298, 295, 294, 292, 290, 288, 287, 283, 282, 282, 276, 275, 275, 274, 273, 273, 272, 272, 271, 271, 269, 269, 268, 267, 267, 266, 263, 263, 262, 262, 261, 260, 259, 259, 259, 258, 256, 255, 254, 254, 254, 253, 253, 253, 253, 252, 252, 252, 252, 251, 251, 250, 250.

The bin size is 1000.
Also given classes. In the above set, the classes are the colors (each color means a class, the colors are black, red, green and blue).
In any bin there can be at most 2 different colors (and the sum of sizes cannot exceed the bin capacity).

Determine the optimal solution (how to pack the items into minimum number of C=1000 capacity bins).

**7. Bin packing with the LIB constraint.**

Given the following items, in the following (fixed) order:

235, 516, 324, 340, 440, 542, 45, 98, 102, 345, 65, 340, 448, 420, 18, 265, 49, 542, 450, 280, 430, 350, 750, 121, 812, 34, 53, 120, 440, 450, 16, 87, 45.

LIB means that „Largest in Bottom”.
We must pack the items in the above order. A bigger item cannot be packed into a bin that already contains a smaller item. For example, we pack the first item (235) into some bin. Then this bin cannot contain the second item (516) since this is larger. The bin capacity is 1000.

Determine the packing with as few bins as possible.

**8. Bin packing with conflicts**

Given the following n=120 items (Falkenauer t120_00 set, see, Unibo homepage)

497, 497, 495, 485, 480, 478, 474, 473, 472, 470, 466, 450, 446, 445, 445, 444, 439, 434, 430, 420, 419, 414, 412, 410, 407, 405, 400, 397, 395, 376, 372, 370, 366, 366, 366, 366, 366, 363, 363, 362, 361, 357, 357, 356, 356, 355, 352, 351, 350, 350, 350, 347, 336, 333, 329, 325, 320, 315, 314, 313, 307, 303, 302, 301, 299, 298, 298, 298, 295, 294, 292, 290, 288, 287, 283, 282, 282, 276, 275, 275, 274, 273, 273, 272, 272, 271, 271, 269, 269, 268, 267, 267, 266, 263, 263, 262, 262, 261, 260, 259, 259, 259, 258, 256, 255, 254, 254, 254, 253, 253, 253, 253, 252, 252, 252, 252, 251, 251, 250, 250.

The bin size is 1000.
Two items are in conflict, if their sum is divisible by 3. In any bin, no two items can be packed if they are in conflict.

Determine the optimal solution (how to pack the items into minimum number of C=1000 capacity bins).

**9. Scheduling couple-of-tasks on a single machine.**

Given the following 12 jobs:

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| a(i) | 1 | 2 | 3 | 4 | 3 | 2 | 1 | 5 | 3 | 1 | 2 | 3 |
| b(i) | 2 | 1 | 8 | 5 | 3 | 1 | 1 | 2 | 3 | 5 | 2 | 5 |
| c(i) | 5 | 4 | 2 | 1 | 3 | 3 | 4 | 1 | 1 | 8 | 2 | 8 |

Any job is determined by a triplet, (a,b,c). It means that the job consist of two tasks. The executing time of the first task is a units, and the execution time of the second task is c units. Between these two tasks, exactly b time units must be spent.
The jobs should be scheduled on a single machine, without preemption. At any time period the machine can process at most one task. (The jobs can be processed in any order.)
Determine a schedule (starting at time zero) for which the makespan is as small as possible (the last task will be finished as soon as possible.

**10. Scheduling on a single machine, to minimize the lateness**

Given 12 jobs with the following data:

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| p(i) | 10 | 2 | 5 | 2 | 1 | 3 | 4 | 6 | 12 | 5 | 4 | 3 |
| r(i) | 2 | 10 | 18 | 34 | 31 | 18 | 14 | 23 | 31 | 5 | 2 | 3 |
| d(i) | 15 | 14 | 27 | 39 | 36 | 25 | 20 | 30 | 50 | 15 | 12 | 8 |

Here p(i) is the processing time of the job, r(i) is the release time, and d(i) the due date.
Any job can be started not before r(i), and (if possible) it should be finished before d(i). If it will not be finished at d(i) but at certain time c(i)>d(i), then c(i)-d(i) is the lateness of the job. The total lateness is minimized (the sum of lateness of all jobs).

**11. Scheduling jobs on unrelated machines.**

Given the following 24 jobs. In the table below it is given the p(i,j) processing times: The execution time for job j is p(i,j), if it is processed by machine Mi.

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M1 | 1 | 8 | 3 | 4 | 3 | 1 | 6 | 5 | 3 | 1 | 2 | 3 |
| M2 | 2 | 1 | 8 | 5 | 2 | 5 | 1 | 2 | 9 | 1 | 4 | 2 |
| M3 | 5 | 4 | 4 | 2 | 8 | 4 | 4 | 6 | 1 | 8 | 2 | 8 |

| i | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 23 | 24 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|
| M1 | 1 | 2 | 3 | 4 | 3 | 2 | 2 | 5 | 2 | 1 | 2 | 3 |
| M2 | 3 | 1 | 8 | 9 | 3 | 1 | 4 | 3 | 3 | 6 | 3 | 1 |
| M3 | 5 | 6 | 6 | 1 | 3 | 3 | 4 | 1 | 1 | 8 | 2 | 8 |

Any machine can process at most one job at any time, preemption is not allowed. Determine the optimal schedule, minimizing the makespan (the finishing time of the last job should be as small as possible).

**12. Set covering**

Given the following subsets of {1,2,…,30} and their weights.

S1={1,5,8,9,12,15}, w1=5
S2={1,2,3,4,5,6,8,9,15,16,17,19}, w2=23
S3={1,4,5,6,9,15,22,23,25,27,28}, w3=21
S4={3,4,5,6,7,9,14,16,23,29}, w4=48
S5={2,4,6,8,10,12}, w5=23
S6={2,3,4,6,8,10}, w6=6
S7={4,5,6,8,9,13}, w7=4
S8={5,6,7,22,23,24,25,26,28}, w8=20
S9={6,7,9,11,15,17,30}, w9=15
S10={7,9,11,13,17,22}, w10=7
S11={8,10,12,14,16,20,24}, w11=8
S12={9,10,11,12,14}, w12=21
S13={10,11,12,13,14,15,16,17,18,19,20}, w13=47
S14={21,22,23,25}, w14=40
S15={22,23,25,28,29,30}, w15=21
S16={23,24,25,26,27,28}, w16=23
S17={24,25,28,29,30}, w17=25
S18={25,26,27,28}, w18=19
S19={28,29,30}, w19=7
S20={29,30}, w20=2

Any subset also have a weight, written as wi above.
The goal is to select some of the subsets so that they „cover" the ground set {1,…,30} (i. e. any number among 1 and 30 is an element of at least one chosen subset), and the total weight of the chosen subsets is as small as possible.

**13. Hypergraph coloring**

Given the following subsets of {1,2,…,30}.

S1={1,5,8,9,12,15},
S2={1,2,3,4,5,6,8,9,15,16,17,19},
S3={1,4,5,6,9,15,22,23,25,27,28},
S4={3,4,5,6,7,9,14,16,23,29},
S5={2,4,6,8,10,12},
S6={2,3,4,6,8,10},
S7={4,5,6,8,9,13},
S8={5,6,7,22,23,24,25,26,28},
S9={6,7,9,11,15,17,30},
S10={7,9,11,13,17,22},
S11={8,10,12,14,16,20,24},
S12={9,10,11,12,14},
S13={10,11,12,13,14,15,16,17,18,19,20},
S14={21,22,23,25},
S15={22,23,25,28,29,30},
S16={23,24,25,26,27,28},
S17={24,25,28,29,30},
S18={25,26,27,28},
S19={28,29,30},
S20={29,30},

Give a color to any element of the ground set (i. e. to any integer between 1 and 30) so that the colors of the elements of each subset are different. Use as few colors as possible.

**14. Minimizing the weighted lateness**

Given 5 jobs, A, B, C, D, E. The execution times of these jobs are 15, 8, 3, 2, 6, respectively. The jobs must be executed on a single machine. The machine can only process one job at a time. Preemption is not allowed (processing of a job must not be interrupted). The machine starts working at time 0. Each job has a due date (a time by which it would be good to have the job completed). These due dates are 19, 23, 28, 15, 32, respectively. If the work is not completed on time, a penalty will be payed. The amount of the penalty: The delay multiplied by a job-specific constant, which is 1, 2, 5, 3, 2 per job. That is, if, for example, the third job is not completed at time t = 28, and for example 2 units are delayed (ends at 30), then 5 times 10 = 10 monetary units is the amount of the penalty because the lateness of this job. Schedule jobs so that the total penalty is minimal.

**15. Packing with conflicts, again**

There are two cities, A and B. Both places have passengers who want to go to the other city. So whoever is in A goes to B, whoever is in B goes to A. The trip is solved by a minibus, a minibus runs between the two cities and goes back and forth. The minibus can accommodate 10 people at a time (apart from the driver). Passengers can be divided into 5 categories: a, b, c, d, e. Some of these “can’t stand each other”, e.g. fans of different competing football teams. These must not be put on a bus at the same time.

The conflicts are specified in the following matrix:

| | a | b | c | d | e |
|---|---|---|---|---|---|
| a | | X | | X | X |
| b | X | | X | | X |
| c | | X | | X | |
| d | X | | X | | |
| e | X | X | | | |

That is, e.g. ‘a’ hates both ‘b’ and ‘d’ and ‘e’, but ‘a’ doesn’t hate ‘c’.

City A has the following number of passengers from each category:

| a | b | c | d | e |
|---|---|---|---|---|
| 18 | 42 | 23 | 8 | 18 |

City B has the following number of passengers from each category:

| a | b | c | d | e |
|---|---|---|---|---|
| 23 | 19 | 15 | 40 | 15 |

Organize the trips in as few rounds as possible.

**16. Open end bin packing**

Given objects, their sizes and cardinalities are as follows:

| size | 23 | 22 | 21 | 19 | 17 | 13 | 12 | 11 | 10 | 7 | 6 | 5 | 4 | 3 | 2 | 1 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| # | 5 | 4 | 2 | 1 | 3 | 6 | 4 | 2 | 3 | 1 | 9 | 4 | 1 | 2 | 3 | 2 |

So, for example, there are 5 items with a size of 23. These items are packed in bins. Each bin has a capacity of 50 units. We want to pack the items in a minimum number of bins, but so that the total size of the items packed in a bin can only exceed the capacity of the bin by a little. This is meant as follows: In each bin, the total size of the objects can be no more than the capacity of the bin, plus the size of the smallest item packed in the bin.
So, for example, if we put two objects of size 23 in the bin, it will fit because $23 + 23 \leq 50$. If we put another object of size 22 here, that's fine too, because $23 + 23 + 22 \leq 50 + 22$.
But for example, if we put the following in a bin: 21, 17, 16, 3, that’s not good because $21 + 17 + 16 + 3 = 57$, which is greater than $50 + 3 = 53$. Pack he items into as few bins as possible.

**17. Scheduling with rejection**

Given jobs. Each job i has a processing time p(i) as well as a penalty u(i). The following table contains the data:

| i | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| p(i) | 58 | 47 | 63 | 91 | 43 | 56 | 85 | 24 | 12 | 53 | 57 | 75 | 23 | 45 | 12 | 45 | 32 | 33 | 12 | 35 | 54 | 23 |
| u(i) | 14 | 23 | 54 | 2 | 5 | 6 | 80 | 72 | 58 | 42 | 33 | 12 | 6 | 2 | 1 | 8 | 10 | 4 | 6 | 5 | 6 | 50 |

The jobs are processed with 3 machines. It is also possible to refuse to process a job, in which case we must pay the penalty assigned to the job. The machines start working at time 0, any machine can only process one job at a time. The completion time of the last job completed is called “makespan”. The goal is to minimize the makespan + total penalty. So for each job, decide whether to process it with one of the machines or not. If so, with which machine. Then the makespan plus the value of the total penalty is obtained, we want to minimize this.

**18. TSP with time windows**

A time-windowed version of the Traveling Salesman Problem.
Given 8 cities, A, B, C, D, E, F, G, H. The agent must visit these in turn (in some order). The agent leaves city A and returns here after the tour; the order of the other cities is not predetermined. The distances (or costs) between the cities are as follows:

| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| A | X | 1 | 8 | 9 | 3 | 2 | 4 | 6 |
| B | 1 | X | 7 | 4 | 1 | 3 | 9 | 2 |
| C | 8 | 7 | X | 6 | 8 | 2 | 4 | 5 |
| D | 9 | 4 | 6 | X | 9 | 3 | 1 | 1 |
| E | 3 | 1 | 8 | 9 | X | 4 | 2 | 7 |
| F | 2 | 3 | 2 | 3 | 4 | X | 6 | 3 |
| G | 4 | 9 | 4 | 1 | 2 | 6 | X | 5 |
| H | 6 | 2 | 5 | 1 | 7 | 3 | 5 | X |

For the cities, the time windows (the agent must arrive between min and max) are as follows.

| | A | B | C | D | E | F | G | H |
|---|---|---|---|---|---|---|---|---|
| min | 0 | 3 | 32 | 15 | 3 | 6 | 25 | 50 |
| max | 100 | 12 | 40 | 24 | 6 | 10 | 30 | 52 |

To be solved:

A, Is there any feasible solution to the problem? (That is, is there an order for the agent to leave city A, visit the cities (each exactly once), and then return to city A so that arrives in cities B,…, E between the earliest and latest times specified?) It is assumed that you spend 0 time in each city.

B, If there is no allowed order, then let be a penalty of 1 unit per city if the agent arrives in the city outside the time window. Define the tour so that the total penalty is as small as possible (i.e., as few cities as possible violate the time window condition).

C, Let the cost of the trip be the length of the trip plus the amount of penalties. (So, for example, if we go through the cities in 48 units of time and come back, plus there are, say, two time-window violations, the total cost is 48 + 2 = 50.) Let's minimize that cost!