The University of Manchester
Alliance Manchester Business School

Coursework

BMAN73701
Python Programming for Business Intelligence and Analytics
2025-2026

Lecturers: Ahmed Kheiri, Xian Yang
GTA: Qian Zhao

Last update: 26/10/2025



Due date: 2pm, 14th January, 2026

Submission: The electronic versions of your **report (saved as a single .pdf file)** and your **code (compressed as a single .zip file)** are to be submitted through Canvas.

Format: Conduct the coding and analysis using appropriate functionalities provided in Python (ideally version 3.12 or later). Summarise your work using Microsoft Word (or an equivalent word processor). All main results must be included in your report, as further detailed in the brief below. Keep your writing concise, around **4000 words**, excluding figures, tables, references, and appendices. Figures and tables must be numbered and referred to in the text. Include your Python code as an appendix within the report (e.g., by printing the code to PDF and appending it to the final report PDF). You must convert your Word report to a PDF before submission. In addition, submit a separate .zip file containing all your source code files. There is no limit on the length of your code. For the report, use a minimum font size of Arial 11pt. For the code, use any readable font with a minimum size of 11pt. Hand-written submissions will NOT be accepted.

While the brief below is broken up into questions/tasks, this is done to help you in guiding your analysis and understanding what tasks must be undertaken. It does not determine the organisation of your report. The report should not quote the questions/tasks in the brief. It should be a coherent piece of work, where you describe the problem at hand and you explain the steps taken to analyse it and solve it. It should contain an introduction section and a conclusion section.

Assessment: The final coursework mark will count 70% to your overall module mark; the remaining 30% will come from the on-campus, paper-based, closed-book exam. Ensure that the material presented in your report is correct and reproducible, and that the Python code is correct and readable. In terms of professional presentation, pay

attention to things such as layout, font size, keeping to the page limit, spelling, grammar, use of tables and charts, consistent number of decimal places.

Each group MUST submit their report by the deadline indicated above. Late submission will be penalised in accordance with the School's guidelines on late submission (see the Assessment & Feedback page on Canvas). Each group MUST work on its own. Any evidence of plagiarism will result in a mark of zero.

Each group must keep a record of their meetings, in particular, date, meeting duration, who attended the meeting and the actions assigned to each member of the group after the meeting. This record should be added as an appendix to the group report. Insufficient contribution to groupwork will result in a mark of zero.

Questions: For any questions related to the coursework, please pose your question in **the Coursework discussion board** available on Canvas. This ensures that all students have access to the same guidance and clarifications.

Brief: The University of Manchester operates a powerful mainframe computer that supports research activities across multiple sectors, including engineering, social sciences, and healthcare. Eleanor Mitchell, the director of the computer facility, is responsible for ensuring that it runs smoothly at all times. As the new academic year begins, Eleanor faces a complex challenge. She must schedule her team of student operators, who monitor and maintain the mainframe, taking into account their different availabilities, skills, experience levels, and wage rates. Some students can only work on certain days, and each has a minimum weekly commitment. Eleanor wants to optimise the schedule to minimise total labour costs while ensuring a fair distribution of hours and adequate coverage of critical skills.

The mainframe has been employed in collecting and processing data from a local Accident and Emergency Department (AED) in Manchester. The department must cope with variable patient workloads while meeting the government target that 95% of patients leave within 4 hours. Over a 4-week period, data were collected on 8,781 patients, including their age, arrival time, length of stay, treatments, investigations, and whether they breached the 4-hour target.

Using Python, your task is to optimise the schedule of student operators using a linear programming model, and analyse a sample of the AED data, gathered through systems managed on the mainframe, to help hospital management understand the workload, patient flow, and factors contributing to breaches or extended stays.

To complete your assignment, Eleanor has provided you with the following data and information.

All operators are current University of Manchester students, including four bachelor's students (E. Khan, Y. Chen, A. Taylor, and R. Zidane) and two master's students (R. Perez and C. Santos). They have limited daily availability, as shown in the table below:

Operator	Mon	Tue	Wed	Thu	Fri
E. Khan	6	0	6	0	6
Y. Chen	0	6	0	6	0
A. Taylor	4	8	4	0	4
R. Zidane	5	5	5	0	5
R. Perez	3	0	3	8	0
C. Santos	0	0	0	6	2

The operators have different wage rates based on their experience and programming ability:

Operator	Wage Rate
E. Khan	£25/hour
Y. Chen	£26/hour
A. Taylor	£24/hour
R. Zidane	£23/hour
R. Perez	£28/hour
C. Santos	£30/hour

Bachelor's students have a minimum weekly commitment of 8 hours, while master's students have 7 hours, ensuring adequate familiarity with mainframe operations. The mainframe operates from 8am to 10pm, Monday through Friday, with one operator on duty during these hours. On weekends, other staff cover operational shifts.

The challenge is to optimise the operator schedules to meet minimum weekly commitments while considering availability, wage rates, and operational hours. Due to budget constraints, Eleanor wants you to develop a linear programming model to minimise total labour costs by determining the number of hours to assign to each operator on each day, while ensuring fair distribution of hours and coverage of critical skills.

In addition, you are provided with `AED4weeks.csv`, collected via the mainframe, containing anonymised data from 8,781 patients who attended a local Accident and Emergency Department over a 4-week period. You will need to create a random sample of 400 patients from this dataset to help hospital management describe the nature of the workload, explore patterns, and investigate potential causes of breaches or prolonged stays. The dataset include:

- **ID:** patient ID;
- **Age:** of patient in years;
- **Day:** numbers the days 1 to 28;
- **DayofWeek:** Monday $\dots$ Sunday;
- **Period:** patient's time of arrival at AED, where '0' means between midnight and 1am, '1' means between 1am and 2am, etc.
- **LoS:** Patient's length of stay in the AED (in minutes);
- **Breachornot:** patients who spend more than 4 hours in the AED have breached the NHS 4-hour performance target;
- **HRG:** Health Related Group - this is a code that categorises the treatment that the patient received;
- **noofinvestigations:** number of diagnostic tests and other investigations that were performed on the patient;
- **nooftreatments:** number of treatments that the patient received;
- **noofpatients:** the number of patients who were already in the AED when the patient arrived.

Task 1: Eleanor has provided you with the data on student operators, their daily availability, wage rates, minimum weekly commitments, and operational hours of the mainframe. Using Python, implement the linear programming model to determine the optimal allocation of hours for each operator per day, minimising total labour costs.

Task 2: Analyse the initial schedule generated by your model and note any inequities in hour distribution. Then, re-model and optimise the schedule under the following scenarios:

> **Scenario i:** Achieve a fairer distribution of hours while allowing up to 1.8% increase in total labour cost.
>
> **Scenario ii:** Achieve the fairest possible distribution of hours and determine the minimum overall cost increase required.

Provide clear explanations of the resulting schedules.

Task 3: Incorporate the skill-based dimension. Each operator has at least one skill ("Programming" or "Troubleshooting") and Eleanor wants at least 6 hours of operation per skill each day. Reformulate the LP problem to meet this requirement. Determine whether a feasible solution exists and report the associated cost increase if applicable.

Operator	Programming	Troubleshooting
E. Khan	✓	
Y. Chen	✓	
A. Taylor		✓
R. Zidane		✓
R. Perez	✓	
C. Santos	✓	✓

Task 4: Draw a random sample of 400 patients from the dataset `AED4weeks.csv` (set a unique Python random seed). Describe the main features of the sample as if presenting to hospital management. Include:

- Main features of individual variables.
- Interesting relationships between variables.

You should include Python numerical and graphical output.

Task 5: Using your sample, investigate potential factors contributing to breaches or prolonged stays.

Task 6: Apply suitable machine learning methods to predict whether a patient will breach the 4-hour target in the AED. Justify your choice of method and interpret the results in a way that is meaningful for hospital management, highlighting key factors associated with breaches and supporting your conclusions with appropriate metrics and visualisations.

Task 7: Develop the Python code for submission, which will not be directly assessed via the report. The code should be well-structured, thoroughly commented, intuitive to use, generalisable for any parameter values or datasets, and flexible, allowing the user to access all analyses performed in the previous tasks. The program should run without errors and make sensible use of the programming techniques covered in the course. Break the code into small, meaningful functions, follow good programming practices and employ object-oriented programming concepts where appropriate. Provide a user interface that enables interaction, for example: choosing to display the overall cost, the number of hours assigned to a particular operator, the schedule on a specific day, the schedule of a specific operator, the overall schedule, statistics from the AED dataset, etc. Users should be able to navigate through options, view results, and continue or quit the program as desired.

An example of a user interface is as follows:

```
> Choose one of the following options:
> 1: Show the overall cost
> 2: Show the number of hours assigned to an operator
> 3: Show the schedule on a particular day
> 4: Show the schedule of an operator
> 5: Show the overall schedule
> 6: Show the stats/plots of a variable
> . . .
> Q: Quit the program
1
> The overall cost is £20000
> Do you want to continue (Y/N)?
Y
> Choose one of the following options:
> 1: Show the overall cost
> 2: Show the number of hours assigned to an operator
> 3: Show the schedule on a particular day
> 4: Show the schedule of an operator
> 5: Show the overall schedule
> 6: Show the stats/plots of a variable
> . . .
> Q: Quit the program
2
> Enter the name of the operator:
E. Khan
> The number of hours assigned to E. Khan is 19 hours
> Do you want to continue (Y/N)?
N
> Good bye!
```

Task 8: Extend the functionality of your program to allow more interactive data management. Users should be able to input a patient's ID to retrieve all corresponding information, specify minimum and maximum values for a particular variable to identify patients falling within that range, and perform actions such as deleting or modifying data. Additionally, implement a log file that records user actions within the system, providing a clear audit trail of interactions and modifications. Ensure that all extensions are intuitive to use, properly integrated with the existing user interface, and maintain the program's robustness and flexibility.

Hints:

(a) Implement all steps in Python. Use Python to perform statistics, generate tables, plots, and implement optimisation or machine learning models. Do NOT use Excel or any other software.

(b) Use different Python files to implement independent steps and tasks. You should decide on the best way to divide the tasks between Python files.

(c) Breaking the tasks into individual files or modules allows work to be distributed among team members and reduces computational load.

(d) All steps must be fully reproducible without editing the Python files. If the dataset is replaced with a new one, the scripts should execute in order and produce updated results automatically.

(e) You are expected to explore and research methods as needed. For this coursework, you may use Python packages such as Numpy, Pandas, Matplotlib, Scikit-learn, and PuLP. You should consult the documentation of the relevant package to understand functions and methods required for your analysis.

(f) Follow good programming practices: document your code thoroughly, use meaningful variable names, break the code into reusable functions, and employ object-oriented programming concepts where appropriate.

(g) Ensure that the user interface is intuitive and flexible, allowing interaction with the data and results for both scheduling and AED analysis tasks.

(h) Begin early to allow sufficient time to complete all tasks. Leaving work to the last minute often results in poor submissions and lower marks.

(i) Make regular back-up copies of your work (hard drive, USB, cloud storage). Loss of work due to technical problems will not be accepted as a reason for an extension.

Coursework Marking Scheme

Your coursework mark will be determined based on the quality of both the report and the code. The final coursework mark will count 70% of your overall module mark. The remaining 30% will come from the on-campus, paper-based, closed-book exam.

Ensure that the material presented in your report is correct, well-explained, and reproducible. The Python code must also be correct, readable, and submitted separately as a .zip file. If the Python files are not submitted, you will receive zero marks allocated for Python.

Each group MUST submit their report by the deadline indicated in the coursework brief. Late submissions will be penalised in accordance with the School's guidelines on late submission (see Assessment & Feedback page on Canvas). Each group MUST work independently. Any evidence of plagiarism will result in a mark of zero.

Marking Breakdown:

- **Task 1:** Linear programming model for optimal scheduling – 10%
- **Task 2:** Schedule fairness and cost optimisation – 10%
- **Task 3:** Skill-based LP model – 7%
- **Task 4:** AED data sampling and descriptive analysis – 15%
- **Task 5:** Investigation of breach factors – 8%
- **Task 6:** Machine learning model for breach prediction – 10%
- **Task 7:** Program structure and user interface – 20%
- **Task 8:** Program extensions (data handling and logging) – 10%
- Professional presentation and report clarity – 10%
    - Pay attention to layout, font size, keeping to the page limit, spelling, grammar, use of tables and charts, consistent number of decimal places.

Note that all plots, tables, schedules, and outputs from all tasks must be accessible through the user interface.