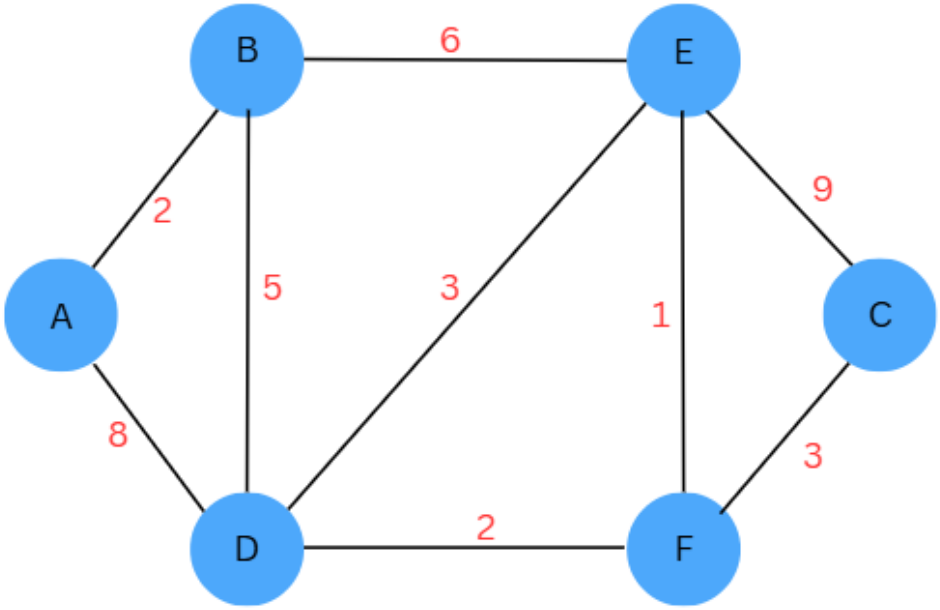
Algorithms

MAP:



Done on Canva

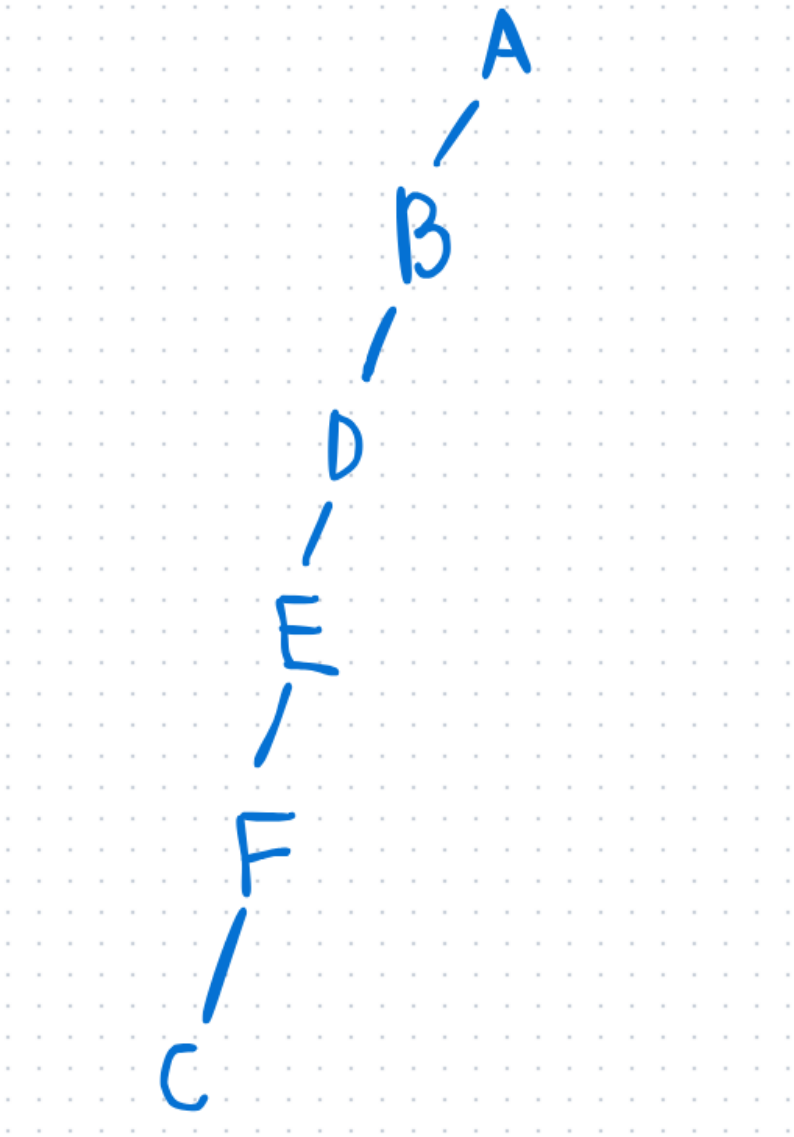
Now DFS traversal order:

Starting point-A

Ending Point-C

A -> B -> D -> E -> F -> C

Diagram(tree):



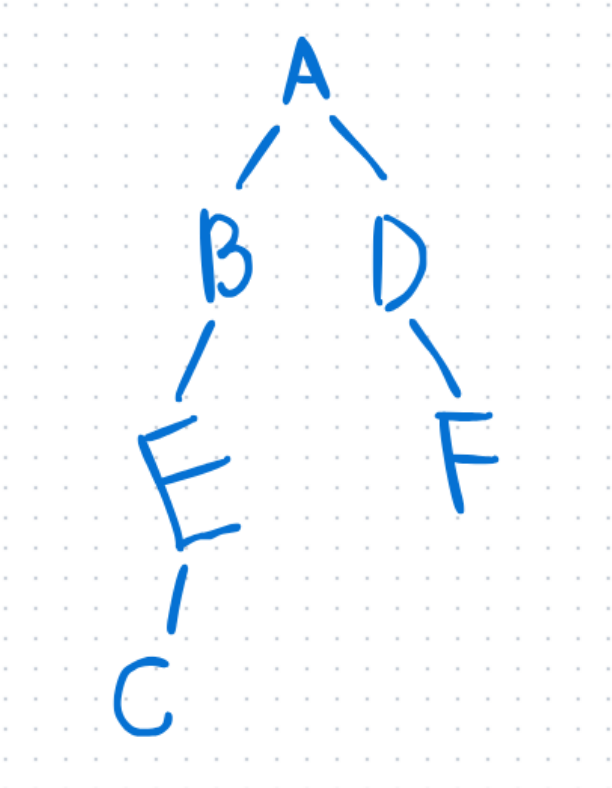
Now DFS traversal order:

Starting point-A

Ending Point-C

A -> B -> D -> E -> F -> C

Diagram(tree):



Dijkstra's Algorithm:

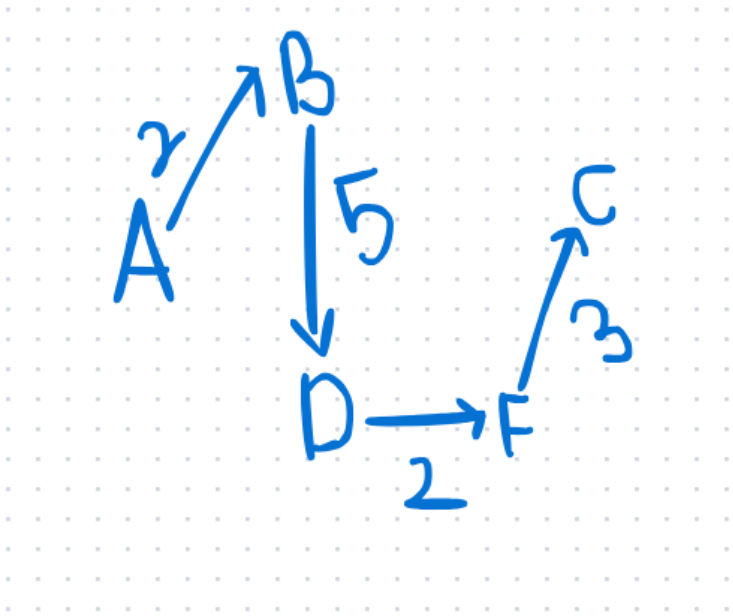
The Shortest Path:

Starting point- A

Ending Point- C

A -> B -> D -> F -> C

Diagram:



A* Heuristic function:

LET,

h(A) = 7.5

h(B) = 5.5

h(D) = 5.2

h(E) = 2.5

h(F) = 1.5

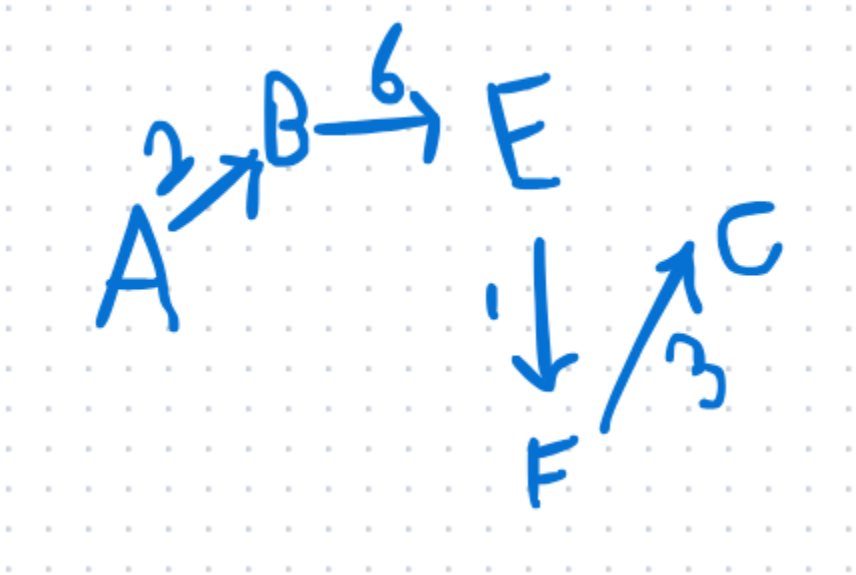
h(C) = 0

Starting point- A

Ending point- C

Path found- A-> B -> E -> F -> C

Shortest path- 12



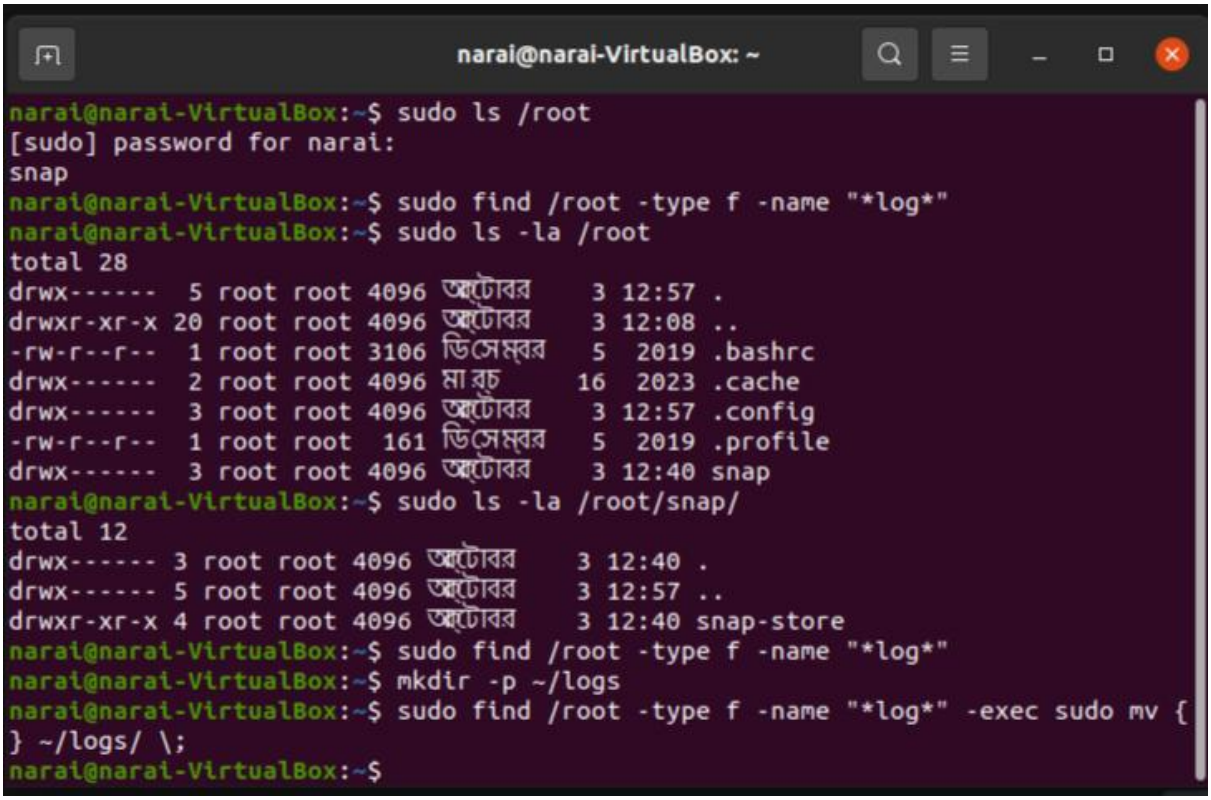
Practical:

Algorithm:

Greedy method

1. At first, we start from the point (x0,y0)
2. We find the shortest Euclidean path from that point
3. We add that point to the sorted list
4. We repeat this process until all the points are visited
5. We reverse the list at the end

Unix terminal



Why I did what I did.

1. I attempted to list all the contents of /root directory to verify if it exists and to grant my user sudo privileges
2. Snap directory exists in /root
3. I searched the entire /root directory structure to find any files with ‘log’ word in it.
4. No file like that exists in the /root structure
5. I listed all the files in /root with detailed information
6. I examined snap directory and its contents
7. No log-file exists
8. I repeated my search to see if I missed any files
9. No output
10. I created destination directory as required

Machine Learning and Object Detection

Section 1: Core ML Understanding

1. Machine Learning is when computers learn or are trained to learn from data to make decisions or predictions, instead of following strict programmed rules like in

traditional programming. A "model" is the pattern or rules the machine learns from the data.

2. **Supervised Learning:** The computer learns from labeled examples where we show it both the input and the correct output. For example, if we show it the picture of a bird or multiple images of a bird, it will be able to classify the features is a group and can gradually teach itself to identify birds in general

   Unsupervised Learning: The computer finds patterns in data without any labels or answers provided, like grouping similar things on its own.

   Reinforcement Learning: The computer learns by trial and error through rewards and punishments. If it succeeds in making a correct assumption, its reward system goes up and vice versa, and it gradually gets the hang of it, and identifies the problem and its solution by itself.

   For rover, all three could be used, we can show multiple pictures of the mars yard, rocks, using supervised learning

   Unsupervised Learning can be used to classify various elements together by itself without much human intervention saving costs,

   Reinforcement Learning would work best - it could learn optimal navigation by trying different paths and getting rewards for efficient movement and punishments for collisions.

3. a. Overfitting happens when the model fits the training data too well but doesn’t know how to deal with real-world data. For example, if the Mars rover were trained to perfectly identify Mars rock based on a limited or similar dataset, it might fail to identify an oddly shaped rock because it just isn’t like the training data.

We train a very simple model to distinguish between "rock" and "soil." If the model is too basic, it might fail to learn the difference altogether, performing poorly even on the training images it learned from. This is underfitting.

b. Data preprocessing is the cleaning and organizing of dataset before we feed it into our AI model. Raw data can be messy, hence via certain processes they are ‘cleaned’ up and made ready for the model, for example, resizing, normalization, formatting, etc.

c. Data augmentation is a technique to artificially expand the size of our training dataset by creating slightly modified versions of existing data. It’s useful when we don’t have a lot of data to work on. Different types of Data augmentations are geometric transformations and color code transformations

d. When building a machine learning model, we need a way to check how well it will perform on brand-new, unseen data. It’s called data splitting.

We randomly divide our entire dataset into three separate groups:

1. **Training Set:** The largest portion. It learns all its patterns from this data.
2. **Validation Set:** A smaller portion. We use this set to "tune" the model's settings and check its performance during training without letting it see the final test.
3. **Test Set:** Another portion. This is the **final exam**. After the model is fully trained and tuned, we use this set exactly once to get an unbiased estimate of how it will perform in the real world.
4. Qualities of a good data-set:

Large & Diverse Volume: It needs thousands of images, not hundreds. It must cover all scenarios so the model can generalize aside from memorizing

Accurate & Consistent Labeling: The labels must be correct and applied the same way across all images.

Balanced Classes: No single category should dominate.

High Relevance & Quality: The images must be from the same environment as the real-world task (e.g., actual Mars-like terrain, not Earth deserts) and be clear, in-focus, and high-resolution.

5. **a. Model Inferencing:** This is the process of using a trained machine learning model to make predictions on new and unseen data. In order to measure a model's performance and accuracy we test it using test dataset and compare the model's predictions against the known, correct answers.

   b. i. **mAP@50-95:** An overall score for how well the model finds and labels objects, when it's graded on drawing perfect, tight boxes around them

   ii. mAP@50: An overall score for how well the model finds and labels objects, even if it isn't always perfectly drawn boxes.

   iii. Recall: How good a model is at finding everything

   iv. Precision: How good a model is at avoiding false predictions/alarms.

   c. A model is good enough if its mAP score meets the project's needs and the balance between precision and recall is alright.

   d. A confusion matrix is a simple table showing the kinds of mistakes the model is making. It signifies the types and

counts of the model's correct predictions and its specific mistakes, showing what exactly is confusing.

Section 2: Understanding the Dataset

1.

First image: low resolution, not a clear image, not properly annotated, blurry, and dark

Second image: The image isn’t focused on the arrow too much background clutter, is not annotated, and has lighting inconsistencies

Third image: Doesn’t have many problems, but has lighting inconsistencies

Fourth image: pretty accurate

Fifth image: Poor framing, inconsistent annotation, too much background clutter.

2. Documentation of roboflow:

1. after creating the project Bird detection I cloned images from Kaggle’s dataset

2. After cloning the annotation was done

3. after the annotation, I applied augmentation, The augmentations that were applied are:

- **Rotation:** -15° to +15°

- **Brightness:** -20% to +20%
- **Shear:** ±10°
- **Flip:** Horizontal
- **Blur:** Up to 1px
- **Noise:** 1-2%

The link : https://app.roboflow.com/first-project-gxc5e/bird-detection-2ghva/1

Section 3: Object Detection Model Training

1. Link: https://www.kaggle.com/code/nazifaanjum/notebook84a13f9d67
2. **a.** The confusion matrix shows that:
    - The model detects mallet very well
    - The model detects bottle fairly well
    - Some mallets and bottles are wrongly detected as background
    - Some misclassifications happen between bottle and mallet

    Yes I see some problems

    The problem is that the model sometimes confuses the objects as background. And misses detection

It happens for having unclear images with poor annotation and unclear object-background contrast, and also for having imbalanced data(more mallet images than bottles)

To solve the problem, there should be a more balanced number of images and images in diverse settings. We can add more augmentation and increase training epochs. Retraining could be an option.

b. The values show that:

- The model has high precision and recall, so it can accurately detect most objects and makes fewer false predictions
- mAP@50 value is very good, meaning it has string detection accuracy at a basic IoU threshold
- mAP@50–95 is lower, meaning the model is not as accurate when a stricter IoU threshold is applied.
- Bottle is detected slightly better

c. Overall, the model performs very well in detecting bottles and mallets, but the model could perform better if it were trained with more varied images, using augmentations to avoid false detections

Section 4: Reasoning and Debugging of Practical Problems

1. **a.** The issue here is that the model assumes blue to be the main indicator of the bottle; it could happen if there are more images of blue coloured bottles, causing an

imbalance. It is a problem with the dataset: dataset bias and class imbalance.

b. Adding diverse-coloured images of bottles in the dataset, data augmentation, and fine-tuning with varied data.

2. a. Spurious correlation or shortcut learning

b. The model uses colour as a shortcut and uses colour as an indicator of mallet**.** Primarily, a dataset problem. Dataset bias.

c. We can add more shape variety of mallet images, and different coloured mallet images as dataset, we can apply colour augmentation. Adding orange non-mallet images in the background.

3 a. dataset imbalance and lack of background (negative) samples.

b. Adding many images with only background without any mallet in it and labelling them as background. Clearly annotating every image properly with objects on Mars, along with empty backgrounds, then the model can distinguish between real objects and empty scenes.

Github link: https://github.com/na-rai/230041155_software_recruitment/tree/main