Numerisches Lösen - Bisektion

1. **Vorgehensmodell**

   Als Vorgehensmodell habe ich Kanban gewählt, da es eine klare visuelle Darstellung des Projektfortschritts ermöglicht und gleichzeitig hohe Flexibilität bietet. Das es besonders einfach zu implementieren ist kann es besonders in diesem kleinen Umfang schnell eingebaut werden.

2. **Programmier-Paradigma (Strukturiert, Objektorientiert)**

   Als Programmier-Paradigma habe ich mich für das Objektorientierte Programmierung entschieden um den Code möglichst übersichtlich, lesbar und modularisierbar zu gestallten.

3. **Zeitmanagement**

Erste Berechnungen in Excel, um Bisektion zu verstehen (Aufgabe 1 & 2)	3 Stunden
PAP zu dem Bisektionsproblem erstellen (Aufgabe 3)	1 Stunde
Planungsarbeiten (Aufgabe 4)	2 Stunden
Programmierung sowie Tests in Python (Aufgabe 5)	2.5 Stunden
Alternative Lösungsmöglichkeit (Aufgabe 6)	3 Stunden
Diagramme mittels matplotlib (Aufgabe 7)	1.5 Stunden
Polynom Berechnung (Aufgabe 8)	1.5 Stunden
Elektrische Leitung Berechnung (Aufgabe 9)	5 Stunden
Summe	**19.5 Stunden**

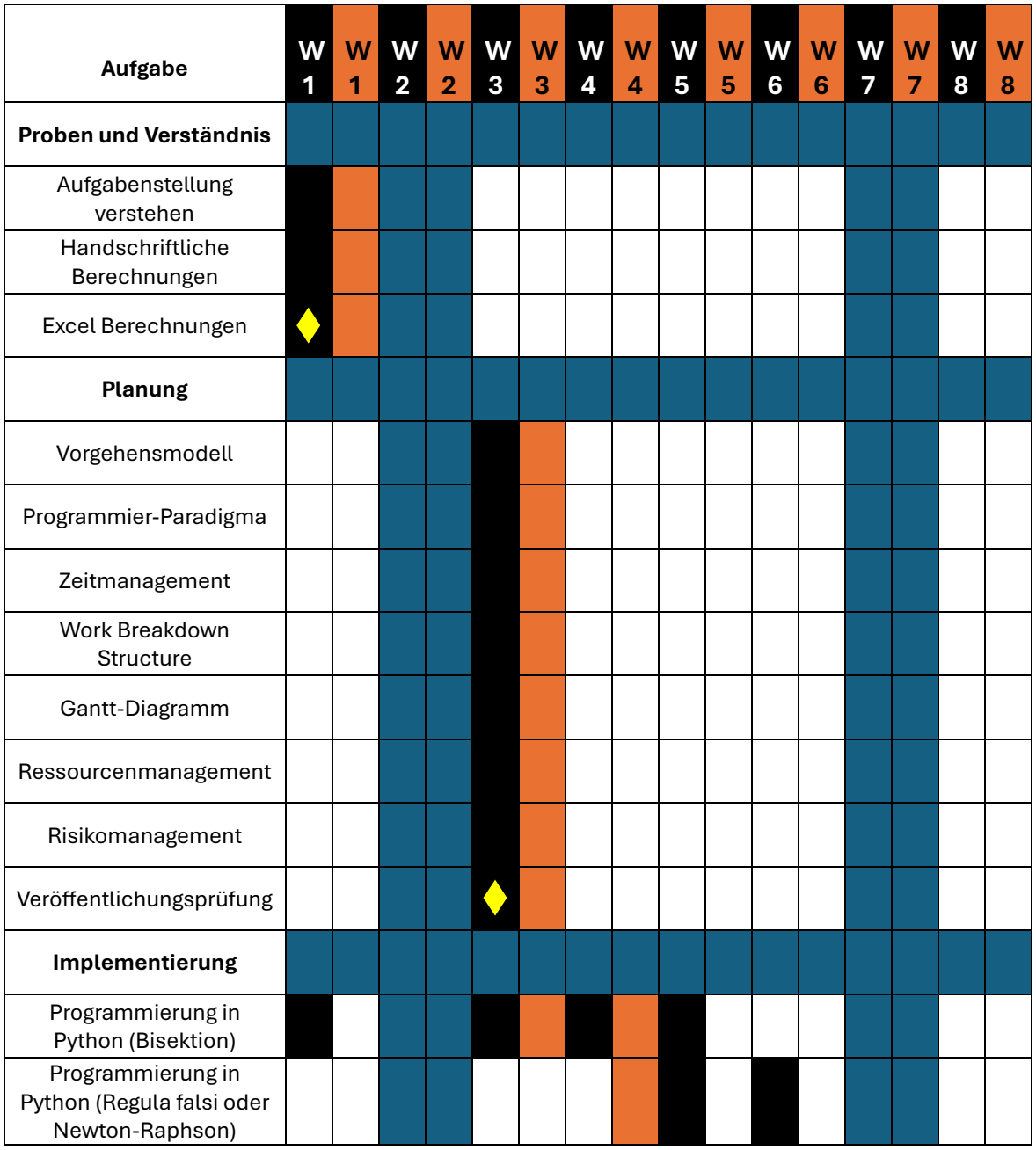
4. **Work Breakdown Structure (WBS)**
   a. Proben und Verständnis
      i. Aufgabenstellung verstehen
      ii. Handschriftliche Berechnungen
      iii. Excel Berechnungen
   b. Planung
      i. Vorgehensmodell
      ii. Programmier-Paradigma
      iii. Zeitmanagement
      iv. Work Breakdown Structure
      v. Gantt-Diagramm
      vi. Ressourcenmanagement
      vii. Risikomanagement
      viii. Veröffentlichungsprüfung
   c. Implementierung
      i. Programmierung in Python (Bisektion)
      ii. Programmierung in Python (Regula falsi oder Newton-Raphson)
      iii. Matplotlib Programmierung
      iv. Polynom Programmierung in Python

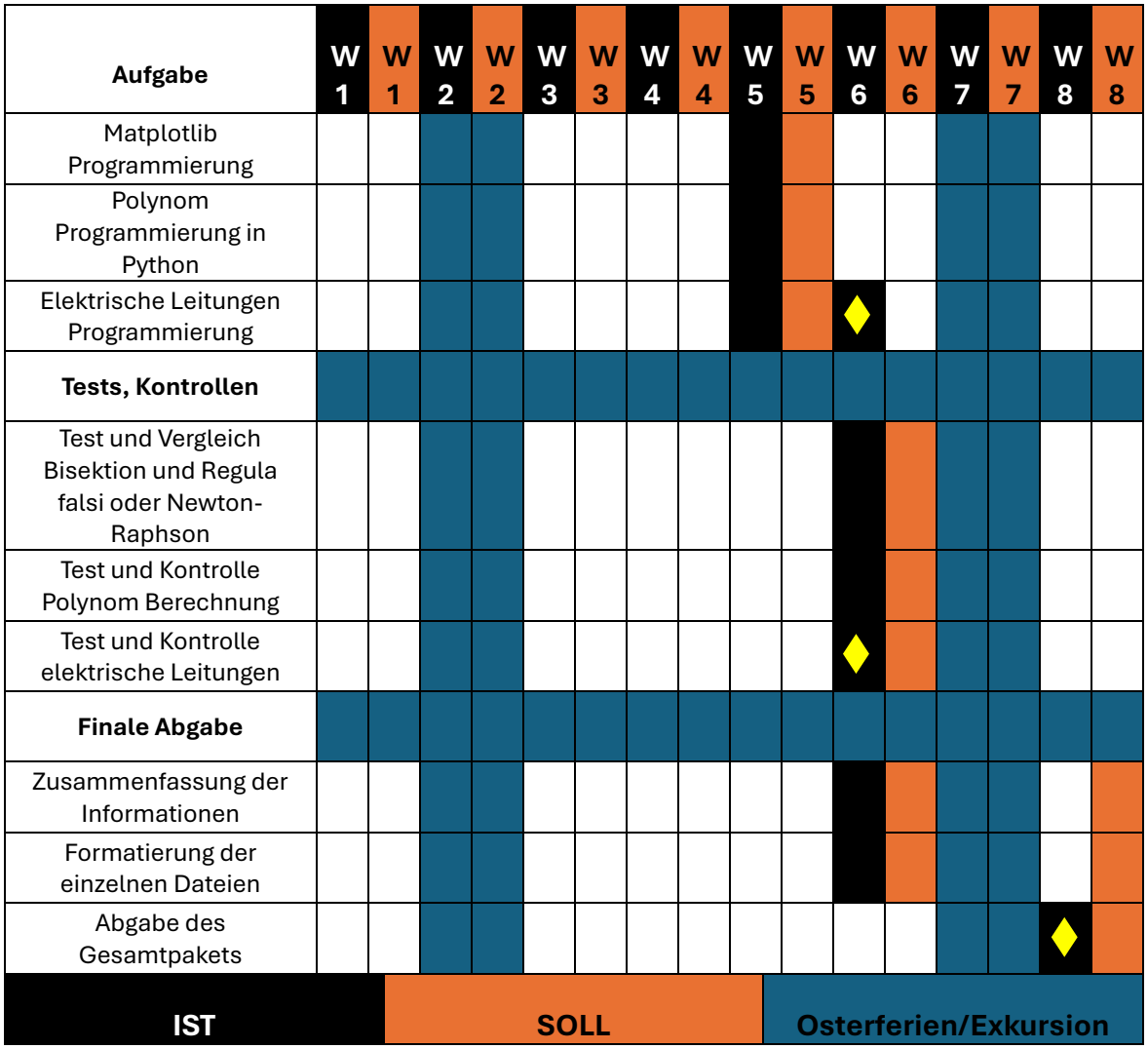
v. Elektrische Leitungen Programmierung in Python

d. Tests, Kontrollen
   i. Test und Vergleich Bisektion und Regula falsi oder Newton-Raphson
   ii. Test und Kontrolle Polynom Berechnung
   iii. Test und Kontrolle elektrische Leitungen Berechnung

e. Finale Abgabe
   i. Zusammenfassung der Informationen
   ii. Formatierung der einzelnen Dateien
   iii. Abgabe des gesamt Packets

5. Gantt-Diagramm





♦ **... Meilensteine**

1. Abschluss der Proben
2. Abschluss der Planung
3. Abschluss der Implementierung
4. Abschluss der Tests und Kontrollen
5. Finale Abgabe

6. **Ressourcenmanagement**
    a. Personal:
        i. 1 Entwickler (Ich)
    b. Hardware
        i. Laptop
        ii. Ladegerät
        iii. Maus
        iv. Papier
        v. Stift
        vi. Taschenrechner

   c. Software
      i. Python
      ii. Matplotlib
      iii. Word
      iv. Excel
      v. Google

7. Risikomanagement

   a. Interne Risiken
      i. Unterschätzer Aufwand -> Pufferzeit einplanen
      ii. Speicherverlust -> Cloud-Speicher / GitHub-Repo
      iii. Hardwareprobleme -> Ersatzhardware und Cloud-Speicher
      iv. Fehlerhafter Code -> Stack Overflow / Google / Professoren bei Fragen
   b. Externe Risiken
      i. Blackout in Österreich -> Zeitverlust
      ii. Software Ausfälle -> Wechsel auf andere Software
      iii. Umweltkatastrophen -> Ausfall von Arbeit im Gesamten

8. Veröffentlichungsprüfung

   a. Matplotlib: Open-Source-Bibliothek -> Mit Python unter PSF-Lizenz
   b. Visual Studio Code (VSC): Open-Source-Editor unter MIT-Lizenz
      i. Verbreitung: Matplotlib und Python können frei verwendet, verändert und weitervergeben werden, solange die Lizenzbedingungen der Software eingehalten werden. Da VSC nicht mitgeliefert wird fällt die Lizenz hierbei weg.