# ENGR1510J Recitation Class

## Week 4

Su Qijian

UM-SJTU Joint Institute

Communication, Computer & Programming, MATLAB scripting
October 5, 2024

# Presentation Overview

# Reminders

- RC will be held only on **Thursday** from **20:30 to 22:00**.
- Recording will be provided.
- RC will only review playbooks and solve the worksheet, **no new content** will be introduced.
- We will focus on **hard questions** from the playbook to optimize RC time. Ensure you are **familiar with all playbook answers** for exam preparation.
- RC will only give ideas on how to solve the worksheet (e.g., pseudo-code), but **NO direct answers** will be provided in the RC.
- A makeup RC for **Week 3** will be provided **tomorrow from 20:30 to 22:00**. The makeup RC will be **online**, and a recording will also be provided in case it conflicts with your schedule.

# Canvas

- The official resource platform.
- Everything related to assignment notifications, timeline announcements,
- Documents, quizzes, and surveys will be posted on **Canvas**.
- Set up your Canvas notifications and settings properly.
- Check course files.
- You just need to submit your regular homework and projects through **Gitea**, no need to submit them on Canvas unless it tells you to do so.

# Mattermost

- The official instant message platform.
- For general discussions and short questions.
- Channels each have their own purpose and offer guidance in their description headers.
- Asking questions in the **Course Question** channel and the individual channel is more advised.
- No screenshots if they are not necessary.
- Do not post your code in any public channel; provide a link to your code instead.

# Wechat

- Not official communication tools used in **ENGR1510J**.
- Direct requests on **WeChat** will be ignored.
- Please contact us using tools other than **WeChat**.

# Email

- Used in the case of a serious project/homework related concern or personal issue that needs to be discussed privately.
- Use brackets **[ENGR151]** and then include your full name in the subject line of the email.

# Gitea

- Your homework and projects should be uploaded to your **Gitea repository**.
- Set up your **Gitea account** as well as your **SSH key**.

# Git Workflow

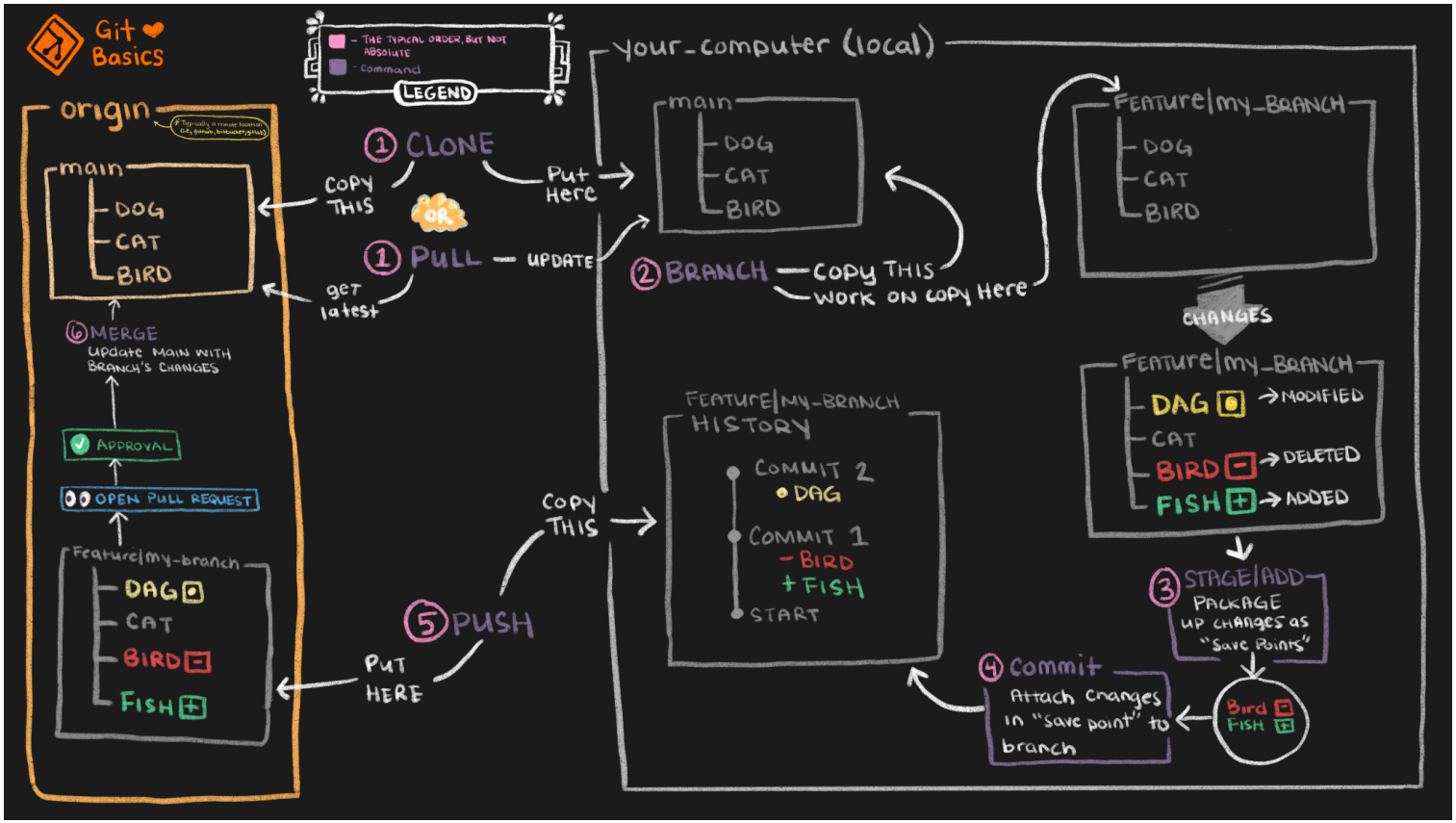



Figure: Git Workflow.[1]

[1] Source: https://www.getdbt.com/analytics-engineering/transformation/git-workflow

# Commits

- Commits should be used for saving meaningful changes in the project.
- Avoid force commits or using `git push --force`, as this can overwrite work done by others. A deduction will be applied if you force push without an appropriate reason.
- Follow the provided file structure format in the submission.

# Issues

- Open an issue in **Gitea** if you encounter any code-related problems.
- Remember to tick the checkbox in issues once you finish the task of each milestone (e.g. your project).
- Close the issue once you finish the task or the issue is resolved.

# Pull Requests

- Follow the instructions stated in the course support wiki.
- Remember to review and comment on the pull requests and your teammates' code.
- Please do not contain any Chinese characters in both your commits and pull request comments.
- Approve pull requests as soon as possible if everything is ready (review, comments, debugging). Emergency situations often arise, especially near deadlines!

# Releases

- Remember to create a release for each submission or else a heavy deduction will be applied.
- Use the correct release tag for each release.

# Playbook Review

**Disclaimer:** The answers provided here are not guaranteed to be correct. Please use them as a reference only and verify with reliable sources.

# Review: c1

**Q1: What is Linux?**

- Linux is an open-source, Unix-like operating system kernel that forms the core of many operating systems.

**Q2: Why is Linux important when studying CSE (Computer Science, Computer Engineering, and Software Engineering)?**

- Linux is widely used in industry, especially in server environments and development.
- Many programming tools, such as compilers, debuggers, and version control systems, are built around or perform better in Linux.

# Review: c1

**Q3: Can a device do both input and output?**

- Yes, such as touchscreen or network cards.

**Q4: What are the CPU and the RAM?**

- **CPU (Central Processing Unit):**
  - It performs calculations and executes instructions to run programs.
  - It consists of two main components: the arithmetic logic unit (ALU), which performs mathematical operations, and the control unit, which directs operations within the computer.
- **RAM (Random Access Memory):**
  - RAM is a type of computer memory that stores data and machine code currently being used.
  - It is volatile, meaning that it only retains information while the computer is powered on.

# Review: c1

**Q5: What is the usage of a programming language?**

- A programming language is a formal language used to write instructions for a computer to execute.

**Q6: Why do we need to be very specific when writing an algorithm?**

- Being specific ensures that the algorithm covers all possible cases and handles edge conditions effectively.
- Clear, specific instructions help other developers understand and maintain the code.

# Review: c1

**Q7: What is the difference between interpreted and compiled languages?**

- **Interpreted Languages:**
  - Code is executed line by line by an interpreter at runtime.
  - No need for a separate compilation step.
  - Slower execution due to real-time interpretation.
- **Compiled Languages:**
  - Code is converted into machine code by a compiler before execution.
  - Faster execution because the code is precompiled into a binary format.
  - Requires a separate compilation step before running the program.

# Review: c1

**Q8: What are ”high-level” and ”low-level” programming languages?**

- **High-Level Languages:**
  - Closer to human languages, abstracting away most of the hardware details.
  - Easier to read, write, and maintain.
- **Low-Level Languages:**
  - Closer to machine code, providing little to no abstraction from the hardware.
  - More efficient in terms of execution but harder to understand and write.

# Review: c1

**Q9: What is MATLAB good at?**

- MATLAB is highly efficient for numerical computations, data analysis, and algorithm development, making it ideal for engineering, scientific, and mathematical applications.

**Q10: Why should the input and output always be clearly defined?**

- It avoids ambiguity, making it easier to verify correctness and troubleshoot errors.
- Clear definitions improve communication and collaboration between developers, helping ensure that the program meets user requirements.
- It also helps in maintaining code and facilitating future updates or modifications.

# Review: c1

**Q11: What is the difference between a mathematical and a physical solution?**

- **Mathematical Solution:**
  - A solution derived purely from mathematical models and equations.
  - It may involve idealized assumptions that simplify the problem for analysis.
- **Physical Solution:**
  - A solution that is physically realizable and accounts for real-world constraints like friction, air resistance, or material properties.
  - It reflects actual conditions and may differ from idealized mathematical models.

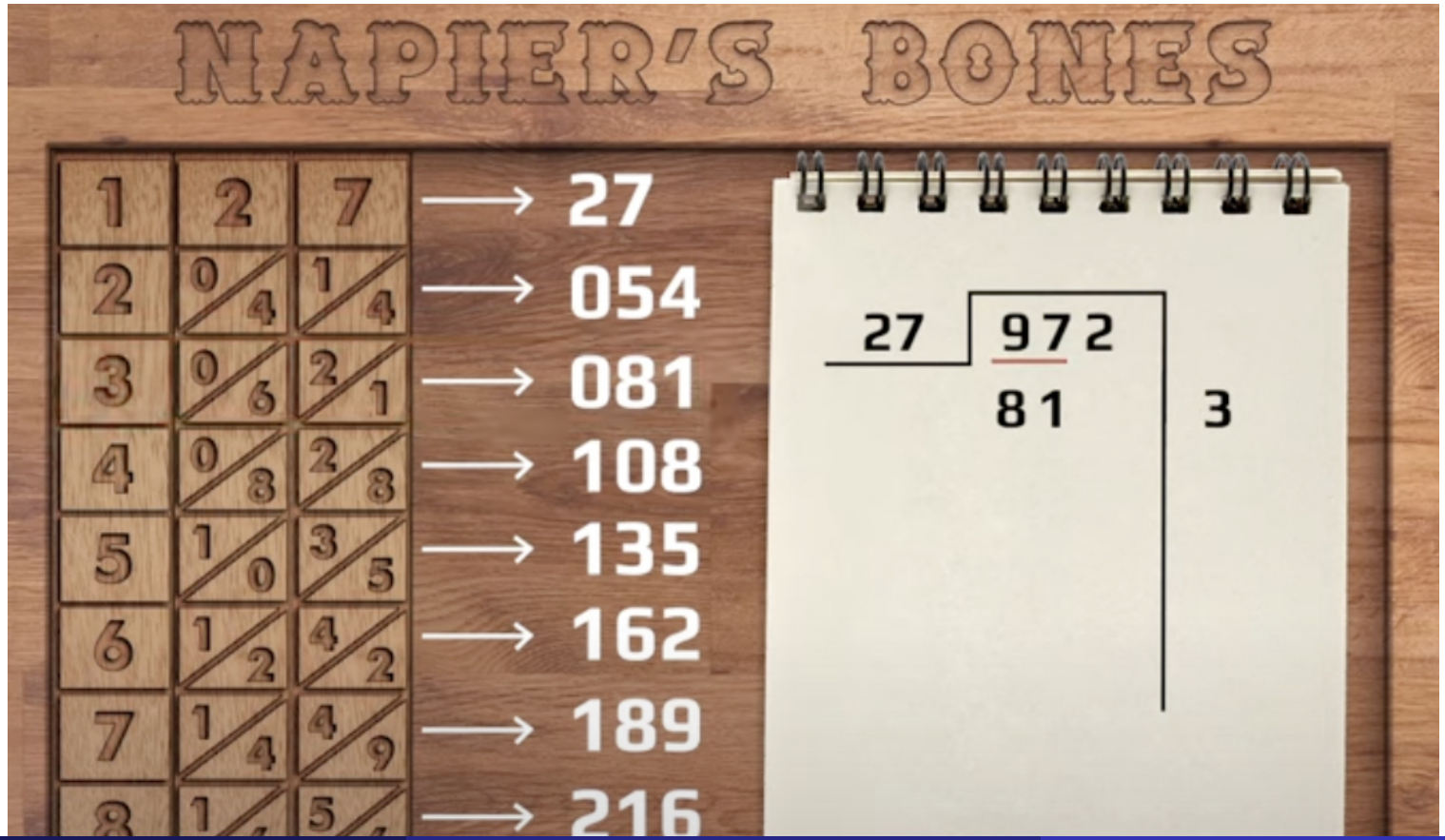
# Worksheets Review

**Note:** For some of the questions, we won't directly provide the entire source code. Instead, we will offer ideas, a piece of code, or pseudo-code to help guide you on the worksheet questions.

# Review: w1

**Napier's bones is a method invented by John Napier to simplify multiplication. Below is the algorithm for using Napier's Bones:**

- Detailed explanation: [Link].
- A video demonstration: [Link].



# Review: w1

**Input:** digit $x$ from 2-9, integer $n$
**Output:** $x \times n$

1. Arrange Napier's bones such that the first row corresponds to $n$.
2. Write down the rightmost digit in the corresponding row.
3. Add the two adjacent numbers and the carry digit in the same row to the left.
4. Write down the ones place digit of the result from step 3.
5. Carry the tens place digit from the result of step 3.
6. Repeat steps 3-5 until you reach the leftmost column.
7. Add the carry digit to the leftmost digit and write down the final result.

# Review: w1

Linux is an open-source, Unix-like operating system that is widely used across various platforms such as servers, desktops, and embedded systems. Refer to:

- [Link]

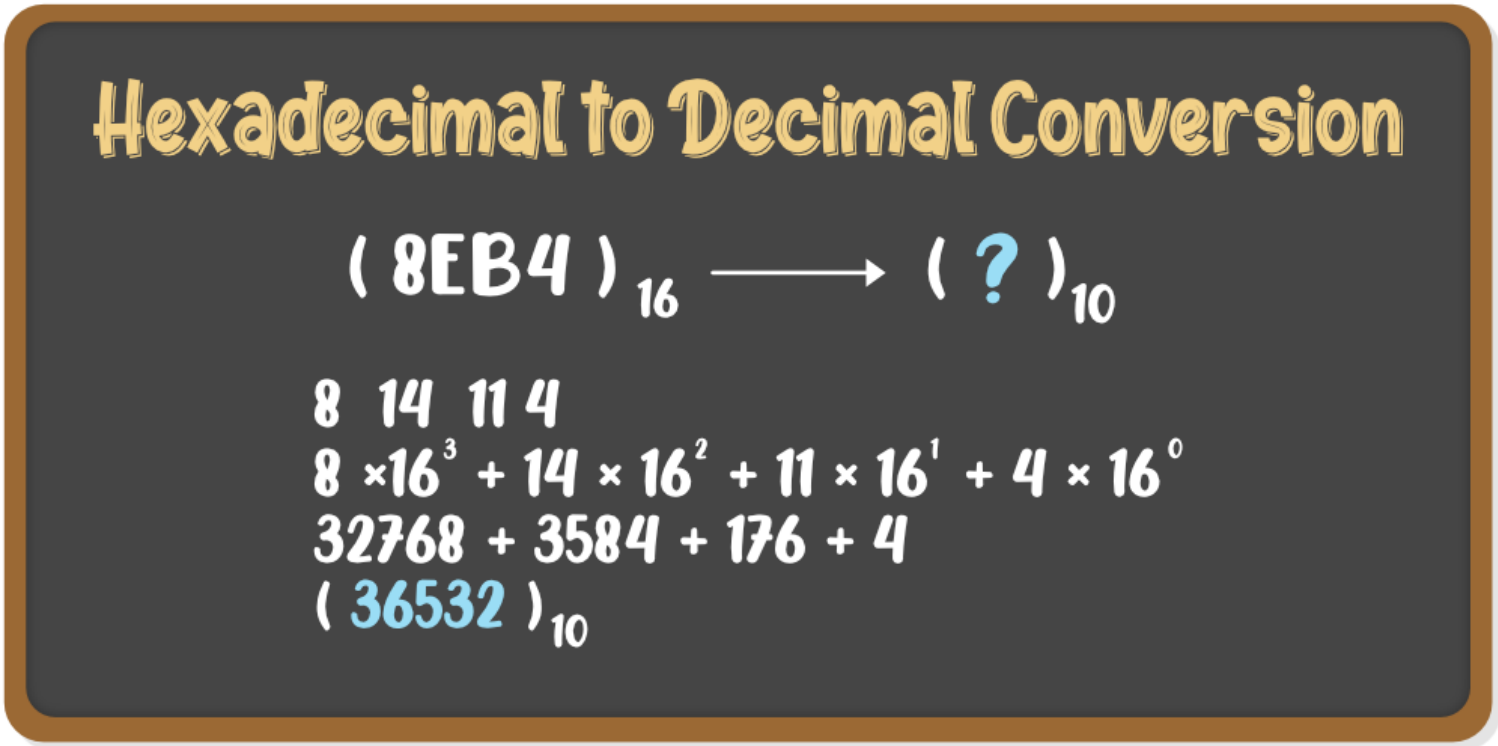
The von Neumann architecture, is a computer architecture model that uses the same memory for storing both instructions and data. It consists of a central processing unit (CPU), memory, and input/output (I/O) units. The CPU retrieves instructions and data from memory, processdes them, and then sends results back to memory or I/O devices. Refer to:

- [Link]

# Review: w1

Number system conversions are the process of converting numbers from one base to another. Refer to:

- [Link]



# Review: w1

## Decimal to Binary Conversion

$(243)_{10} \longrightarrow (?)_2$

| | | |
|---|---|---|
| 2 | 243 | 1 |
| 2 | 121 | 1 |
| 2 | 60 | 0 |
| 2 | 30 | 0 |
| 2 | 15 | 1 |
| 2 | 7 | 1 |
| 2 | 3 | 1 |
| | 1 | |

$\longrightarrow (11110011)_2$

## Binary to Octal Conversion

$(11101011)_2 \longrightarrow (?)_8$

011 101 011

3 5 3

$\longrightarrow (353)_8$

# Review: w1

**Algorithm: Convert Decimal to Binary**
**Input:** digit $x$
**Output:** $x$ in binary form

1. Write down the remainder of $x$ divided by 2 (write the remainders from right to left).
2. Update $x$: $x \leftarrow \lfloor x/2 \rfloor$ (integer division).
3. Repeat steps 1 and 2 until $x = 0$.

**Example:** Convert 13 to binary.

- $13 \div 2 = 6$, remainder = 1
- $6 \div 2 = 3$, remainder = 0
- $3 \div 2 = 1$, remainder = 1
- $1 \div 2 = 0$, remainder = 1

Reading the remainders from right to left, the binary representation of 13 is $1101_2$.

# Review: w1

**Refer to:**
[Link]

- **Interpreted Language**: Python; Perl; PHP; Bash; Markdown; Ruby; Javascript
- **Compiled Language**: C#; Ada95; O'Caml; Pascal; Fortran; Scala; Erlang
- **Interpreted/Compiled Language**: Haskell; Lisp
- **Compiled then Interpreted Language**: Java
- **Neither**: Assembly

# Review: w2

**Graphical User Interface (GUI)** It is a visual interface that allows users to interact with electronic devices using graphical elements such as icons, buttons, and windows. It is intuitive and easy to use, making it ideal for non-technical users.
**Command Line Interface (CLI)** It is a text-based interface where users type commands to interact with the system. CLI offers greater control, precision, and speed for experienced users, making it a preferred tool for system administrators, developers, and engineers.

# Review: w2

To start MATLAB in ”No Desktop Mode” (headless mode), follow these steps:

1. Open a terminal (CLI) on your system.
2. Start MATLAB in command-line mode using the following command (MacOS and Linux):
   `./matlab -nodisplay`
3. Start MATLAB in command-line mode using the following command (Windows):
   `matlab -nosplash -nodesktop`

# Review: w2

Rewrite the simple program from slide (2.6—2.60) using the 's' flag for the input function. Then convert the inputs from the 's' mode into into numbers.

$$num1 = input('Input\ a\ number :', 's');$$

$$number1 = str2double(num1);$$

# Review: w2

**1. Generate a 10 × 10 matrix A composed of random elements.**

$$A = \texttt{rand(10, 10);}$$

**2. Extract the seventh element on the third row of A**

(i) Using its index:

( index = (row - 1) + (column - 1) $\times$number of rows $+ 1$)

$$A(63);$$

(ii) Using its coordinates:

$$A(3, 7);$$

**3. Delete the third column and the fourth row of A.**

$$A1 = A([1:3\ 5:end], [1:2\ 4:end]);$$

# Review: w2

**4. Extract the sixth row and the second column from A.**

$$row6 = A(6, :);$$

$$col2 = A(:, 2);$$

**5. Extract the 4 × 4 matrix at the center of A.**

$$A2 = A(4 : 7, 4 : 7);$$

**6. Construct the following matrix:**

$$\begin{pmatrix} A & A^T & B \\ A^T & A & C \end{pmatrix}$$

where B is the sum along of the rows of A and A, and C is the subtraction of the sum of the rows of A with the sum of the rows of A.

$$B = sum(A, 2) + sum((A'), 2); disp(B);$$

$$C = sum(A, 2) - sum((A'), 2); disp(C);$$

# Review: w2

**MATLAB Script: Truth Table for AND, OR, and XOR Operations**

```
A = [0 0 1 1];
B = [0 1 0 1];
AND = A & B; % logical AND
OR = A | B; % logical OR
XOR = xor(A, B); % logical XOR

TT = [A; B; AND; OR; XOR]';
disp('A B AND OR XOR');
disp(TT);
% A B AND OR XOR
% 0   0   0   0   0
% 0   1   0   1   1
% 1   0   0   1   1
% 1   1   1   1   0
```

# Review: w2

**ASCII** stands for *American Standard Code for Information Interchange*. It is a character encoding standard that assigns a numerical value to letters, digits, and punctuation, allowing text to be represented in computers.

```
cook@pop-os:~$ ascii -d
  0 NUL    16 DLE    32      48 0    64 @    80 P    96 `   112 p
  1 SOH    17 DC1    33 !    49 1    65 A    81 Q    97 a   113 q
  2 STX    18 DC2    34 "    50 2    66 B    82 R    98 b   114 r
  3 ETX    19 DC3    35 #    51 3    67 C    83 S    99 c   115 s
  4 EOT    20 DC4    36 $    52 4    68 D    84 T   100 d   116 t
  5 ENQ    21 NAK    37 %    53 5    69 E    85 U   101 e   117 u
  6 ACK    22 SYN    38 &    54 6    70 F    86 V   102 f   118 v
  7 BEL    23 ETB    39 '    55 7    71 G    87 W   103 g   119 w
  8 BS     24 CAN    40 (    56 8    72 H    88 X   104 h   120 x
  9 HT     25 EM     41 )    57 9    73 I    89 Y   105 i   121 y
 10 LF     26 SUB    42 *    58 :    74 J    90 Z   106 j   122 z
 11 VT     27 ESC    43 +    59 ;    75 K    91 [   107 k   123 {
 12 FF     28 FS     44 ,    60 <    76 L    92 \   108 l   124 |
 13 CR     29 GS     45 -    61 =    77 M    93 ]   109 m   125 }
 14 SO     30 RS     46 .    62 >    78 N    94 ^   110 n   126 ~
 15 SI     31 US     47 /    63 ?    79 O    95 _   111 o   127 DEL
```

Figure: ASCII Table.[2]

[2] Source: https://www.johndcook.com/blog/2022/05/28/how-to-memorize-the-ascii-table/

# Review: w2

**Algorithm: ASCII Code Lookup**
**Input:** A single character key
**Output:** ASCII code of the character

1. Prompt the user to input a key.
2. Check if the input length is exactly 1 character:
   - If true:
     - Convert the key to its ASCII code using the appropriate function. (eg. `double(key)`)
     - Display the ASCII code.
   - Else:
     - Display an error message indicating the input is invalid.

**Example 1: Valid Input**

- Input: `A`
- Output: The ASCII code of `A` is 65

**Example 2: Invalid Input**

- Input: `AB`
- Output: Invalid input. Please enter a single character.

# Review: w2

**Algorithm: Count Vowels and Consonants**
**Input:** A word entered by the user
**Output:** Number of vowels and consonants in the word

1. Initialize vowel_count = 0
2. Initialize consonant_count = 0
3. Define a set of vowels: {a, e, i, o, u}
4. Prompt the user to input a word
5. Convert the word to lowercase using the `lower()` function
6. For each character in the word:
   - If the character is in the vowels set:
     - Increment vowel_count
   - Else if the character is an alphabetic letter (check using `isletter()`):
     - Increment consonant_count
7. Output the vowel_count and consonant_count

# Review: w2

**Generate a random 10×10 matrix, double all the elements less than 5, triple all the ones between 5 and 10, and set all the others to 0 if they are even or 1 if they are odd.**

1. Generate a random 10x10 matrix with values between 0 and 20:
$$M = randi([0, 20], 10, 10); M0 = M;$$
2. Double all the elements less than 5:
$$M(M < 5) = M(M < 5) * 2;$$
3. Triple all the elements between 5 and 10:
$$M(M >= 5 \ \& \ M <= 10) = M(M >= 5 \ \& \ M <= 10) * 3;$$
4. Set all the even elements greater than 10 to 0:
$$M(mod(M0, 2) == 0 \ \& \ M0 > 10) = 0;$$
5. Set all the odd elements greater than 10 to 1:
$$M(mod(M0, 2) == 1 \ \& \ M0 > 10) = 1;$$

# References


- Manuel. *c1.pdf*. JI Canvas, 2024. [Link].
- Manuel. *c2.pdf*. JI Canvas, 2024. [Link].
- Manuel. *w1.pdf*. JI Canvas, 2024. [Link].
- Manuel. *w2.pdf*. JI Canvas, 2024. [Link].
- Wang, Yuheng. *RC-1.pdf*. JI Canvas, 2023. [Link].
- Wang, Yuheng. *rc*. FOCS JI Gitea, 2023. [Link].

# The End

Questions? Comments?