NP GIT
Python Puzzle Solvers
2024

Organised by Girls In Tech (Ngee Ann Polytechnic)

7 October 2024

Table of Contents

Event Schedule **3**
Event Details **4**
What to Expect 4
Team Allocation 4
Submission Guidelines 4
Scoring and Penalties 5
Prizes & Recognition Information 7
Challenges **8**
Enchanted Beginnings (Beginners) 8
Flag 1 - The Enchanted Veil (20 Magic Points) 8
Flag 2 - Great Enchanter's Concert (25 Magic Points) 9
Flag 3 - Spellbooks (30 Magic Points) 10
Flag 4 - The Wizard's Cipher (10 Magic Points) 11
Flag 5 - Cracking the Dragon's Lock (20 Magic Points) 12
Flag 6 - The Sorcerer's Chaotic Sorting Spell (35 Magic Points) 13
Flag 7 - The Enchanted Spellbook (40 Magic Points) 14
Sorcerer's Trials (Intermediate) 16
Flag 8 - The Enchanted Strings (60 Magic Points) 16
Flag 9 - The Tower's Enigma (100 Magic Points) 17
Flag 10 - The Lost Scrolls of Magic (80 Magic Points) 18
Flag 11 - The Mystic Potion Inventory (70 Magic Points) 19
Flag 12 - Rune of Eternal Decoding (90 Magic Points) 20
Legendary Sorcery (Advanced) 21
Flag 13 - The Rune of Secrets (140 Magic Points) 21
Flag 14 - The Cryptic Conundrum (110 Magic Points) 23
Flag 15 - Celestial Navigation (140 Magic Points) 24
Frequently Asked Questions **25**
Code of Conduct **26**
Future Events & Contact Information **28**
Upcoming Events 28
Keep In Touch! 28
Contact Information 28

Event Schedule

Time	Details
10.00 am - 10.15 am	Registration Open
10.15 am - 10:30 am	Opening Speech by GIT
10:30 am - 10:45 am	Break into Groups - Get to Know Each Other! - Set up Your GitHub to Collaborate!
10:45 am - 2.15 pm	Commencement of Python Puzzle Solvers! - Leaderboard will be displayed with live feed - Teams are allowed to leave room for Lunch but cannot continue the challenge - Last submission at **2:15pm** *Note: No lunch will be provided but there will be snacks available throughout the event
2:15 pm - 3:00 pm	Networking session / Break Evaluation of Pending Submissions
3:00 pm - 3:15 pm	Feedback to Individual Groups
3.15 pm - 3.30 pm	Prize Presentation Closing Remarks
3:30 pm - 4:00 pm	Collection of Feedback from Participants Networking Session

Event Details

Welcome to the **Python Puzzle Solvers**, a thrilling 3.5-hour Python coding challenge designed to test and enhance your programming skills!

What to Expect

Python Puzzle Solvers is a hands-on coding challenge where you'll solve a series of progressively difficult Python puzzles, all set within a magical theme. You will work in teams to tackle questions of varying difficulty levels. The team with the highest number of Points at the end of the event will be crowned the winner.

Team Allocation

If you registered as part of a team, you will be placed in your pre-selected group. Remember, each team must have at least one female member.

If you opt to be assigned to a team, our organizers will announce your team details on the day of the event.

Submission Guidelines

Platform: Microsoft Teams Channel (themed as “Magical Chambers”)

Format: All submissions must be made in a single message.

- Submit solutions as a .zip file containing your .py files.
- **File Naming:** Use the format TeamName_QuestionNumber (e.g., TeamGiT_Flag1).
- Attach the flag to the message in the format: GIT{flag_obtained}

Submission Limit: Your team can make up to 10 submissions per challenge. Each submission must include:

- Flag Obtained
- Source code

Important: If the submission does not adhere to the required format, the question will not be marked, and no Points will be awarded.

Scoring and Penalties

For each challenge, Magic Points are awarded using a capture-the-flag format. Here's how it works:

Maximum Number Of Points Obtainable: 970 Magic Points

Magic Points Awarded: Each challenge has a set number of magic Points. You will receive the full Points if your flag matches the exact expected output.

Penalties:

Error Type	**Description**	**Percentage deducted**
Logical errors	If the **flag is correct**, but the **logic** behind the code is **faulty** or coincidental, leading to the correct flag by chance rather than by correctly solving the challenge. **60% deduction**: Moderate issues where the logic isn't fully sound, but the solution works because of assumptions or specific input conditions (hardcoding). **70% deduction**: Significant logical flaws where the correct flag is produced by coincidence or luck, rather than a fully accurate solution (fluke). **80% deduction**: Severe breakdown in logic, where the correct flag is submitted, but the solution completely misses key aspects of the problem and is almost entirely based on coincidence.	**60-80%**
Wrong flag output	If the submitted flag **does not match the expected output**	**100 %**
Non-functional code	If the code **does not work** (e.g., does not compile, crashes or does not give the expected flag)	**100%**

- **Flag Mismatch**: If the submitted flag does not match the expected output, no magic points are awarded for that challenge. Participants must ensure their flags are formatted correctly and correspond to the challenge requirements.

- **Cheating Penalty**: A deduction of **100 Magic Points** for the entire team if caught cheating. This includes using unauthorized resources or collaborating inappropriately.
- **Hint Penalty**: Participants can request hints, but each hint comes with a deduction of **10 Magic Points** from their total score.

Prizes & Recognition Information

At Python Puzzle Solvers, we aim to celebrate and acknowledge both team and individual achievements throughout the event.

Team Awards:

- **Wizards of the Day:** Awarded to the top team with the highest Magic Points. The ranking will be based on the total Points accumulated from the challenges.
- **Master Sorcerers:** Awarded to the 1st Runner-up team. This recognition is based on their performance and Points earned.
- **Enchanted Apprentices:** Awarded to the 2nd Runner-up team, recognizing their impressive results and effort.

Participation Recognition:

- **Participation:** This will be awarded to all teams and individuals who actively engaged in the challenges, demonstrating commitment and enthusiasm throughout the event.

We look forward to celebrating your achievements and contributions in this magical journey!

Challenges

Enchanted Beginnings (Beginners)

For each challenge in this stage, submit all flags received for each input. If one question has 3 unique inputs, 3 unique flags are expected for submission.

Flag 1 - The Enchanted Veil (20 Magic Points)

No. of hints available: 1

At Gringotts Wizarding Bank, powerful sorcerers store their most precious artifacts at vault 713, protected by a secret code known only to them. However, to keep the code safe from dark wizards, masking spells are being cast on the code to hide all characters with a "*" except for the last 4.

As an apprentice of the keeper, your task is to cast the spell (write out the code) masking_spell(code) that hides all characters with a "*" except for the last 4.
However, if the code has less than 8 characters, the message "Unstable code" would be sent out as the code is too weak to protect the code. If the code ends with a vowel (a/e/i/o/u), "- Dark Magic Detected" would be added to the masked code. Your flag should look like this: GIT{masked_code}

Expected inputs	Data type	Value
1) code	string	AUC8EH
2) code	string	ZY12QHPO94TR
3) code	string	ZY12QHPO94TO

Flag 2 - Great Enchanter's Concert (25 Magic Points)

No. of hints available: 1

In the land of Eldoria, magical creatures from all realms gather to witness the Great Enchanter's Concert. Due to the seating arrangement, creatures sitting behind each other must be tall enough to see the enchanted stage and performance.

Write a function can_see_stage(seats, enchnated_rows) that takes in

- seats, a nested list that represents the rows of units of creatures' heights
- enchanted_rows is a list of row indices with a magical property requiring a special "visibility" condition where each creature in the enchanted row must be at least 2 units taller than the one in front of them.

The program should return True if all creatures can see the stage, False if any creature's view is blocked or any enchanted row fails its special condition.

For example, if the seating arrangement is [[2, 2, 2], [3, 3, 3], [4, 4, 4]] and the enchanted rows are [1], the function will return False. This is because the creatures in the enchanted row (row 1, with heights [3, 3, 3]) are only 1 unit taller than the creatures in the row directly in front of them (row 0, with heights [2, 2, 2]). For the enchanted row to meet the conditions, the creatures should be at least 2 units taller than those in the previous row, which they are not in this case.

Your flag should look like this: GIT{True/False}

Expected Parameters	Data type	Input 1	Input 2
seats	list	[[1,2,3,4],[3,4,5,6],[7,8,9,10],[11,12,13,14]]	[[1,2,3,4],[1,4,5,6],[7,8,9,10],[11,12,13,14]]
enchanted_rows	List	[1,3]	[1,3]

Flag 3 - Spellbooks (30 Magic Points)

No. of hints available: 3

You are a magician librarian in charge of consolidating spellbooks from 2 wizards. Each spellbook is a dictionary where the keys represent the respective magical schools (eg: Potion Magic, Healing Magic) and the values are list of spells associated with each school. Your task is to consolidate and merge the two spellbooks into one comprehensive spellbook.

The spellbooks follow these rules:

- If a magical spell appears in both spellbook1 and spellbook2, combine the list of spells from both spellbooks. Ensure that all spells are included without any duplicates within the combined list.
- If a magical spell is only present in one spellbook, include it fully in the final merged spellbook.

Write a program that merges both spellbooks with the requirements mentioned. The program should return the merged spellbook as a dictionary.

Your flag should look like this: GIT{merged_spellbook}

Expected Inputs	Data type	value
spellbook1	dict	{ "Elemental Magic": ["Fireball", "Lightning Bolt", "Earthquake"], "Healing Magic": ["Heal", "Greater Heal"], "Dark Magic": ["Curse"] }
spellbook2	Dict	{ "Elemental Magic": ["Fireball", "Ice Storm"], "Healing Magic": ["Heal"], "Necromancy": ["Raise Undead", "Life Drain"] }

Flag 4 - The Wizard’s Cipher (10 Magic Points)

No. of hints available: 3

In the mystical land of Eldoria, the Grand Wizard Castor has hidden a powerful spell within a secret message. To keep it safe, he cast a magical cipher over the message, shifting each letter by a number known only to those who can solve his riddle.

Therefore, you must first solve the riddle and then decrypt the message.

The mysterious riddle is:
“What is a number that, when multiplied by itself, equals 100?”

Expected Inputs	Data type	value
message	string	S vyfo Robwsyxo Qbkxqob!

Flag 5 - Cracking the Dragon’s Lock (20 Magic Points)

No. of hints available: 1

In the mystical world of Eldoria, the legendary Dragon’s Treasure is protected by an enchanted lock. The lock’s pin codes are carefully chosen based on an ancient pattern, whispered only in myths. The Elders of the Arcane Council have entrusted you with discovering the possible pin codes to unlock the treasure.

These enchanted pin codes follow the Dragon's magical rules:

1. The pin must be a **4-digit number** divisible by both **5** and **7**.
2. The first digit of the pin must be **odd**.
3. The second digit must be **odd** and **different from the first**.
4. The third digit must be **even**.
5. Each digit must be **unique** (no digit is repeated).

Your task is to write a Python spell (program) that checks all 4-digit numbers (from 1000 to 9999) and applies these ancient conditions to find the valid pin codes

Once you find all valid pin codes, return the flag in the format:
`GIT{number_of_valid_pins}`.

For example, if you find two valid pin codes, the flag would be `GIT{2}`.

Flag 6 - The Sorcerer's Chaotic Sorting Spell (35 Magic Points)

No. of hints available: 2

The Sorcerer has attempted to cast a powerful sorting spell to arrange his collection of **magic crystals** by their power. Unfortunately, the spell contains several chaotic elements, and instead of ordering the crystals correctly, it results in unpredictable behavior. You, as an apprentice wizard, must correct the spell.

The sorcerer has asked for a specific set of rules:

1. The crystals must be sorted in **descending order** of their power.
2. The power of each crystal is represented as a **string**.
3. Any crystal with a **negative power** must be moved to the **end** of the list, while still maintaining the descending order of the positive ones.

Here's the broken spell (code) the sorcerer wrote:

```
def sort_crystals_by_power(crystal_powers):
    positive_powers = [p for p in crystal_powers if p >= 0]
    negative_powers = [p for p in crystal_powers if p < 0]
    sorted_powers = positive_powers.sort()
    final_powers = sorted_powers + negative_powers
    return final_powers

crystal_powers = ["15", "42", "-9", "33", "57", "-23"]
sorted_crystals = sort_crystals_by_power(crystal_powers)
print(sorted_crystals)
```

The **correct output** should be :
`['57', '42', '33', '15', '-9', '-23']`

Once you fix the spell, return the flag as `GIT{first_element_last_element}`, where `first_element` and `last_element` are the first and last elements in the sorted list.

For example, `GIT{12_-34}`.

Flag 7 - The Enchanted Spellbook (40 Magic Points)

No. of hints available: 2

In the mystical realm of Codeonia, an ancient spellbook has been uncovered, containing a series of magical spells. Each spell is associated with different attributes in the spellbook. Your task is to decode a message from the spellbook using the provided dictionary.

Using the nested dictionary provided, decode the message. The dictionary contains two categories of spells: `"spells"` and `"common_spells"`. Each spell has attributes such as `"description"` and `"power"`.

For each spell name in the encoded message:

1. If the spell is found in the `"spells"` category, retrieve its `"description"`.
2. If the spell is found in the `"common_spells"` category, retrieve its `"power"`.
3. If a spell is not found in either category, it is ignored.

Combine these descriptions and powers into a single string, with each attribute separated by a space.

Once you fix the spell, return the flag as `GIT{decoded_message}`

Expected Inputs	**Data Type**	**Value**
Encrypted Message	string	fireball lightning bolt rays explosion

Spells	dictionary	{
		"spells": {
		"fireball": {
		"description": "Burns with the fury of a thousand suns",
		"power": "High"
		},
		"lightning": {
		"description": "Strikes with the speed of thunder",
		"power": "Medium"
		},
		"bolt": {
		"description": "A quick and sharp surge",
		"power": "Low"
		}
		},

		"common_spells": {
		"rays": {
		"description": "A blazing inferno",
		"power": "High"
		},
		"explosion": {
		"description": "A sudden burst",
		"power": "Low"
		}
		}
		}

Sorcerer’s Trials (Intermediate)

Flag 8 - The Enchanted Strings (60 Magic Points)

No. of hints available: 1

In Aethoria, sorcerers’ skills are judged by how well they can calculate and estimate the cost needed for manipulations of inanimate objects. For this trial, a puppet will move in a plane starting from the original point (0,0). The puppet can move toward UP, DOWN, LEFT and RIGHT with a given step. The trace of puppet movement is shown as the following:
UP 5
DOWN 3
LEFT 3
RIGHT 2

The numbers after the direction are steps. Please write a program to compute the distance from the current position after a sequence of movement and original point. If the distance is a float, then just print the nearest integer.
Example: If the following tuples are given as input to the program: UP 5 DOWN 3 LEFT 3 RIGHT 2 Then, the output of the program should be: 2

Expected Input	**Data Type**	**Value**
Directions	A list of strings denoting the direction and the distance	[“UP 40”, “DOWN 2”, “LEFT 18”, “UP 4”, “RIGHT 20”, “DOWN 30”, “LEFT 5”]

Find the distance on the following tuples: Once you find the distance, submit the flag in the format `GIT{distance}`. For example, if the distance is **2**, the flag would be `GIT{2}`.

Flag 9 - The Tower’s Enigma (100 Magic Points)

No. of hints available: 2

In the mystical realm of Aethoria, the Wizard of Reflections has built a tower that manipulates the fabric of space and time. The tower's architecture is based on the principles of coordinate geometry and trigonometry. Your task is to write a Python program that calculates the reflection Points of a beam of light as it traverses the tower's labyrinthine structure.

The Wizard's Tower of Reflections consists of a series of mirrors placed at specific coordinates in a 3D space. The mirrors are represented by a list of tuples, where each tuple contains the mirror's position (x, y, z) and its normal vector (nx, ny, nz). The normal vector indicates the direction perpendicular to the mirror's surface.

A beam of light enters the tower at a given point (x0, y0, z0) with a direction vector (dx, dy, dz). Your program must calculate the reflection Points of the beam as it bounces off each mirror, taking into account the laws of reflection and the tower's geometry.

Expected Input	Data Type	Values
Mirrors	A list of tuples, where each tuple contains the mirror's position (x, y, z) and its normal vector (nx, ny, nz).	mirrors = [((0, 0, 0), (0, 0, 1)), ((3, 4, 0), (1, 2, 0)), ((-2, 1, 5), (0, 1, 0)),]
Beam	A tuple containing the beam's initial position (x0, y0, z0) and direction vector (dx, dy, dz).	beam = ((1, 2, 3), (1, 1, 1))

The output should be a list of tuples, where each tuple contains the reflection point's coordinates (x, y, z) and the corresponding mirror's index. Once you find the answer, submit the flag by entering the absolute values of the tuples in the list, separated by commas in the format `GIT{tuple1_tuple2_tuple3}`. For example, if the answer is **[((0,0,0),0), (1,1,1), (2,2,2)]**, the flag would be `GIT{((0000)(111)(222)}`.

Flag 10 - The Lost Scrolls of Magic (80 Magic Points)

No. of hints available: 3

In the ancient world of Azura, a powerful wizard has lost his magical scrolls. These scrolls contain the secrets of magic and are hidden in a mysterious temple. The wizard needs your help to retrieve the scrolls. The temple has a series of puzzles that must be solved to unlock the scrolls.

You are given a list of n magical artifacts, each with a unique id, magic_Points, and weight. The wizard can carry a maximum of max_weight units of weight. Your task is to select the artifacts that will give the wizard the maximum magic Points without exceeding the maximum weight.

The inputs are the artifacts, which is a list of tuples containing the id, magic Points and weight of each artifact, as well as the maximum weight the wizard can carry. Write a program that calculates the list of ids and the total magic Points of the selected artifacts.

The list of artifacts and max weight are:

Expected Input	**Data Type**	**Value**
Artifacts	A list of tuples containing the id, magic_Points, and weight of each artifact	artifacts = [(1, 10, 2), (2, 20, 3), (3, 15, 1), (4, 30, 4), (5, 25, 2)]
Max Weight	An integer on the maximum weight the wizard can carry	max_weight = 6

Once you find the answer submit the flag in the format `GIT{artefacts_totalPoints}`. For example, if the selected artifacts are 10 and 14 (in ascending order), and the total Points is 400, the flag would be `GIT{1014_400}`.

Flag 11 - The Mystic Potion Inventory (70 Magic Points)

No. of hints available: 2

In the hidden laboratory of the Wizarding Academy, there's a mystical inventory of potions. Each potion is stored with its **potion name** as the key and the **ingredient list** as the value. Unfortunately, the ingredients have been mixed up, and they are no longer associated with the correct potion names.

The head alchemist has tasked you with fixing the potion inventory. Each potion name should be matched with an ingredient such that:

1. **First Letter Match:** The first letter of the potion name should match the first letter of its primary ingredient.
2. **Second Letter Match:** If there are multiple ingredients starting with the same first letter, then use the second letter of both the potion name and the ingredient to make the match.

Your goal is to rearrange the ingredients in the inventory to ensure that each potion name is correctly paired with an ingredient according to these rules.

The flag will be the concatenation of the sorted ingredients in uppercase.

Expected Inputs	**Data Type**	**Value**

potion_inventory	dictionary	{
		"Polyjuice Potion": "Frost Fern",
		"Amortentia": "Fire Lily",
		"Frostbite Elixir": "Phoenix Feather",
		"Elixir of Life": "Aconite Root",
		"Firestarter Potion":"Elderflower"
		}

Flag 12 - Rune of Eternal Decoding (90 Magic Points)

No. of hints available: 3

A powerful rune has been inscribed with magical numbers, but beware—some numbers are cursed, and attempting to decode them will cause failure. Your task is to decode the rune by summing all the valid numbers (including numbers hidden within **nested lists**). Any invalid values, such as cursed words or objects, must be handled using **`try-except`** to prevent the decoding process from breaking.

Rules:

- Only valid **integers** contribute to the sum.
- If an invalid value (such as a string, list, or other non-numeric data) is encountered, it must be caught and skipped using `try-except`.
- The total sum must exactly equal the **target value** to unlock the flag.
- Return the flag in the form `"GIT{target_value}"` if the sum is exactly the target. If the total does not match the target, return `"GIT{Failed to decode the rune because the target is now CURRENT_TOTAL}."`

Expected Inputs	**Data Type**	**Value**
runes	list	[10, 'cursed', 40, 'broken', 20, 'none', 30, ['trick',20, 30]]
target	integer	160

Legendary Sorcery (Advanced)

Flag 13 - The Rune of Secrets (140 Magic Points)

No. of hints available: 3

In the enchanted world of Cryptoria, an ancient spell known as RSA is used to secure magical communications. RSA encryption involves generating two keys: a public key for encrypting messages and a private key for decrypting them. This encryption method relies on prime numbers and modular arithmetic, but this particular magical variant includes additional cryptographic layers.

Enhanced Encryption Process:

1. **Padding Scheme:** Pad each character's ASCII value by adding a fixed value of `1000` to it.
2. **Multiple Rounds of Encryption:** Perform encryption three times using the public exponent to add complexity.
3. **Hash Function:** Apply a simple hash function, which is the sum of the digits of the encrypted value.

The magic spell follows a simple formula:

$$c = (m + 1000)^e \bmod n$$

where:

- *c* is the encrypted value,
- *m* is the ASCII value of the character,
- *e* is the public exponent, and
- *n* is the product of the prime numbers p and q.

Expected Inputs	Data Type	Value
Prime Rune (p)	int	53
Prime Rune (q)	int	97
Message	string	`!L0VEH@RRYP0TTER`
Public Exponent (*e*)	int	65537

Example Output Format:

If the encrypted values are 123, 456, 789, and 101, your flag should look like this:
`GIT{123_456_789_101}`

Note: You don’t need to find the private key—just focus on encrypting the message!

Flag 14 - The Cryptic Conundrum (110 Magic Points)

No. of hints available: 3

A mage has left behind an encoded message involving a complex cryptographic system combining multiple encryption methods. To decode it, you'll need to reverse-engineer the encryption process.

Encryption Process:

1. **Caesar Cipher**: The mage first applied a Caesar Cipher with a shift of 88 to the plaintext.
2. **Vigenère Cipher**: Next, the message was encrypted using a Vigenère Cipher with the key IWANTMAGICPoints.
3. **Special Numbers**: The mage embedded a pair of special numbers, 55.4156 and -1.7059, which hold a significant meaning. Use these numbers to find a keyword, ignoring any spaces and capitalizing the result (eg. THISISTHEKEYWORD).
4. **Atbash Cipher**: The mage employed an Atbash Cipher, which reverses the alphabet (A becomes Z, B becomes Y, etc.).
5. **Columnar Transposition Cipher**: As a final layer, the mage used a Columnar Transposition Cipher with the keyword derived from the special numbers.

You have been given the encrypted message and the values needed. Write a Python function to decode the message. Your flag should look like this: `GIT{decoded_message}`

Expected Inputs	**Data Type**	**Value**
Encrypted Message	string	Cpud lXypuQkaZ@c neoHculetd .ceqjzpanqpzhra
Caesar Shift	int	88
Vigenère Key	string	IWANTMAGICPoints
Special Numbers	-	55.4156, -1.7059

Flag 15 - Celestial Navigation (140 Magic Points)

No. of hints available: 3

The Wizard of the Cosmos has tasked you with developing a celestial navigation system to guide the wizard's fleet of starships through the vast expanse of space. The system must accurately calculate the orientation of the starships in 3D space, taking into account the complex rotations and transformations involved in interstellar travel.

The wizard used a combination of Euler angles and Quaternions to represent the orientation of the starships. You have been given the sequence of Euler angles (α, β, γ) representing the rotations around the x, y, and z axes, respectively, and the initial orientation of the starship in Quaternion representation.

Write a program to derive a Quaternion representing the starship's final orientation after applying the sequence of rotations. Once you solve it, your flag should look like this: `GIT{w_x_y_z}`. For example, if the Quaternion value is (1,2,3,4), your flag should be `GIT{1_2_3_4}`.

Expected Inputs	**Data Type**	**Value**
Euler angles	A list of tuples, where each tuple contains the Euler angles (α, β, γ) in radians.	euler_angles = [(math.pi / 2, math.pi / 3, math.pi / 4), (math.pi / 4, math.pi / 2, math.pi / 3), (math.pi / 3, math.pi / 4, math.pi / 2)
Initial Orientation	A Quaternion representing the starship's initial orientation.	Quaternion (0.707106781, 0.707106781, 0, 0)

Frequently Asked Questions

1) **Can I still participate even if I'm still a beginner?**
   Yes, of course! The challenge will consist of 3 stages, ranging from beginner to advanced. So even if you are still a beginner, there would still be questions for you to be able to solve and earn Points! Although it's a competition, join us for the vibes and experience!

2) **What tools and software are being used for this event?**
   Participants are recommended to use Visual Studio code for this event.

3) **How are our submissions being evaluated?**
   You would be given the appropriate Points if your team can capture the flag correctly. Unless your code correctly shows the flag, no Points will be awarded. Penalties will be applied if your code does not return the expected flag. (see Event Details)

4) **Can we use AI?**
   No, usage of AI sources such as ChatGpt is not allowed. We will be walking around and doing random checks – if caught/spotted using AI sources, a deduction of 100 Points will be implemented for each instance. After 3 instances, your team will be disqualified.

5) **What if we get stuck on the challenges?**
   You can skip the question and return to it later, or you can reach out to us for a hint! Using hints would result in a 10-point deduction, so use them wisely!

6) **Are we allowed to leave the venue to get a bite?**
   Yes, you are allowed to leave the venue at any time of the challenge which will take place from 11 am to 2:30 pm. However, you are not allowed to bring your laptops along with you to ensure fairness. You would also be given a break from 2:30 pm to 3:30 pm. Be sure to return on time at 3:30 pm for the prize presentations!

7) **Can I access online resources such as my notes during the challenge?**
   Yes, you are allowed to access your notes from your courses. You are also allowed to search for Python documentation or use StackOverflow, but usage of AI is not allowed and penalties will be enforced if caught. (Refer to Qn 7)

8) **Where can we submit our answers?**
   Submission of answers would be via MS Teams, where each team would be allocated their channels to submit their codes.

Code of Conduct

We want to ensure that everyone has a positive and enjoyable experience. Please review and adhere to the following Code of Conduct:

Integrity and Fair Play:

- Participants are expected to uphold the highest standards of integrity.
- Sharing answers or collaborating with other groups during the event is strictly prohibited. Each group must independently solve challenges.

Resource Usage:

- While the use of AI-generated content is not allowed, participants are permitted to search the web for information and resources during the event. However, all code and solutions must be original and developed by the participants.

Respectful Behaviour:

- All participants must treat each other with respect and courtesy. Harassment, discrimination, or any form of disrespectful behaviour will not be tolerated. This includes both online and in-person interactions.

Inclusivity and Punctuality:

- Participants are expected to foster an inclusive environment. Respect the diverse backgrounds and perspectives of fellow participants. Punctuality is crucial for the smooth running of events, and participants are expected to adhere to the schedule.
- All members in a team should be allowed to work on and participate in the work together.

Original Work:

- All code and solutions submitted must be the original work of the participants. Plagiarism or unauthorised use of external code is strictly prohibited.

Communication Channels:

- Official communication channels provided by the organisers should be used for any queries, clarifications, or discussions related to the event. Participants are discouraged from using external platforms for event-related discussions.

Equipment and Software:

- Participants are responsible for ensuring that their equipment and software are in good working order. Technical issues will not be considered as grounds for rule violations.

Event Environment:

- Participants are expected to maintain a quiet and focused environment during the event. Disruptive behaviour, background noise, or inappropriate surroundings that affect the event's integrity may result in penalties.

Consequences for Violations:

- Violation of any of the above rules may result in disqualification from the event, loss of Points, or other penalties as deemed appropriate by the organisers.

Note:

By participating in the event, all attendees agree to adhere to the Code of Conduct. Organisers reserve the right to take necessary actions in response to violations.

Future Events & Contact Information

Upcoming Events

There's so much more with NP Girls In Tech! Here are some enriching and engaging events to look out for in the coming academic year.

November 2024: Build-Your-Website Event

- Gear up for our "Build-Your-Website" event. Create a custom website to showcase your personal branding!

December 2024: Christmas Party

- Celebrate the year's end with NP GiT SIG at our Christmas Party. Network, make new friends, and enjoy a meal together!

Aside from this, do join our telegram channel to hear about event updates, networking opportunities, or additional resources by NP Girls In Tech SIG! Do note that most events are open to females, unless stated otherwise.

Keep In Touch!

NP Girls In Tech SIG, advocates for gender equality to join STEM courses. While our focus is to promote women in STEM, we welcome both genders to our events! After all, it takes everybody's effort to bring about societal change and promote a positive environment for both genders in STEM.

Join our Telegram Channel: https://t.me/+uFwhRejVVzpmOTg1
Follow our instagram: @girlsintech.ngeeann

Contact Information

Need help with the challenges? Facing technical difficulties? Having questions? Here are some helplines to contact!

Jia Xuan (President)
Telegram - @xu3an

Sahana (Vice President)
Telegram - @sa_ha_naaa

Valery (Secretary)
Telegram - @valeryvalhi

Enjia (Head of Technology)
Telegram - @wuenjia

Vy (Technology Committee)
Telegram - @vyyyytrann

Wei Ying (Technology Committee)
Telegram - @getanicesleep

Raeanne (Technology Committee)
Telegram - @Evlorasin

Dr Er (NP GIT SIG Advisor)
er_poi_voon@np.edu.sg