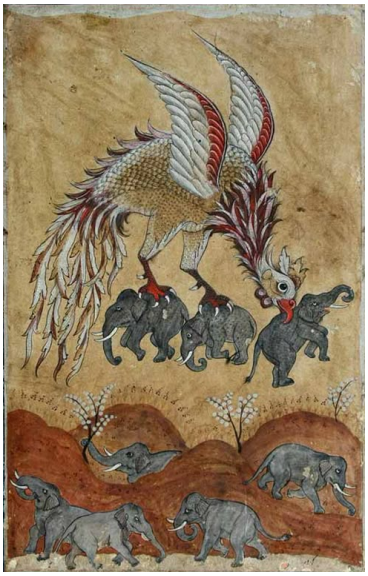
Assignment 1 — The Simurgh's Feather

A text adventure on the road to Mount Qaf

Fakhir Shaheen *(sec - A,B)*

DUE Tue 6 Oct 2026, 11:59 PM **WEIGHT** 3.375% **COVERS** Lectures 1–8 **FILES** 1 **CHECKER** `check.py`



Tilism-e-Hoshruba, eight volumes printed in Lucknow from 1883 to 1893, is one part of the *Dastan-e-Amir Hamza*. Before Hoshruba, the hero Amir Hamza goes to Mount Qaf, the land of jinns, devs and paris. He means to stay eighteen days and stays eighteen years. In the end the Simurgh, a giant bird, helps him home.

This assignment is a prequel. You are the apprentice of Umro Ayyar, the trickster, Hamza's oldest friend, and Umro wants a Simurgh feather. You will write the journey as a text adventure: the game says where you are, you type a command, and a few variables change.

What you will practise

- A `while True` loop that reads a command each time round and stops at `break` (Lecture 7).
- `if/elif/else` checks in the right order (Lecture 5), and Booleans and counters that remember (Lectures 4, 7).
- f-strings (Lecture 3), and print probes to find bugs (Lab 5).

How this assignment works

- Work top to bottom. Read a task, answer its keys, then write its code. Tasks open one at a time: the next opens when this one passes.
- ✎ means pen on this sheet, Shell closed. Only calculators are allowed. ⌨ means at the machine, in `a1.py`.
- The deliverable is `a1.py` in your `a1` folder. Tasks 1 to 10 must PASS, and each is worth the same. Task 11 is a bonus.
- This assignment assumes Labs 1 to 5 and uses Lectures 1 to 8.

Students may NOT look at or copy each other's code, or its structure or logic. Asking seniors or friends for "help" is also not allowed. If caught, a student gets 0 in A1. AI tools (ChatGPT, Claude, Gemini, Copilot, or any other) are NOT allowed. Not for writing code, fixing code or explaining errors. Do not paste this sheet or your file into any AI tool. If you use one in any way, it is plagiarism and you get 0 in A1.

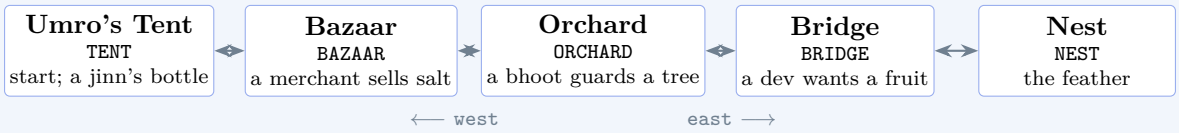
Before you start: get the files

Download the A1 kit from the course calendar and unzip it: `a1.py` and `check.py`, in **one folder**, `a1`. Do not rename them. F5 on `a1.py` plays your game; F5 on `check.py` checks it.

1 Setting Out (~55 min)

The starter runs the game loop. These tasks start the game and fix it.

The road to Qaf



- The `ROAD` block in `a1.py` names the places and prices. Use its names (`BAZAAR`, `SALT_PRICE`), never their numbers: the marking run changes the numbers.
- One command per line, in small letters, from the table below. At the start, and each time you arrive somewhere new, the game prints the place's name, such as `== The Bazaar ==`, then one line about what is there.

Command	Where	What it does	Task
`east`, `west`	anywhere	one place east, one place west	2
`look`	anywhere	says again where you stand and what is there	given
`purse`	anywhere	says how many coins you carry	3
`steps`	anywhere	says how many steps you have taken	4
`get salt`	the bazaar	buys one bag of salt for `SALT_PRICE` coins	5
`rub`	Umro's tent	rubs the jinn's bottle: a wish of `WISH_COINS` coins	6
`get pomegranate`	the orchard	picks one pomegranate, if you carry salt	8
`pay`	the bridge	gives the dev your pomegranate, once	9
`get feather`	the nest	takes the feather: you win	10
`map`	anywhere	draws the road in one line	11, bonus
`quit`	anywhere	prints `You give up the quest.` and ends the game	given
anything else	anywhere	prints `I don't understand that.`	given

Each command is one whole line. `get` on its own, or with any other word after it, is "anything else".

TASK 1 The Prologue

EASY `a1.py` ~15 min

⌨ The game begins by reading your name and the coins Umro gives you.

Your task: Read the name and the coins. Print Umro's words, then the purse line.

Input: line 1: your name. Line 2: the coins, a whole number, 0 or more. Then one command per line, ending with `quit`.

Output: two lines, then the lines of the given loop.
Line 1: `Umro Ayyar: "N, bring me a Simurgh feather."`, where N is the name.
Line 2: `Coins in your purse: C`, where C is the coins.

Sample 1 (the checker uses this one)

Input
```
Sara
9
quit
```

Output
```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 9
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
You give up the quest.
```

Why: Sara goes inside Umro's words, and 9 after the colon. The loop then says where you start, and `quit` ends the game.

Hint: wrap this f-string in single quotes. Read the coins with `int(input())`.

Pen on this sheet, Shell closed, before you write Task 1 in `a1.py`. Only calculators are allowed.

Key 1.1 The user inputs `Hina`, then `5`. Which first line should the program print?
(a) `Umro Ayyar: Hina, bring me a Simurgh feather.`
(b) `Umro Ayyar: "{name}, bring me a Simurgh feather."`
(c) `Umro Ayyar: "Hina, bring me a Simurgh feather."`

Key 1.2 The user inputs `Hina Baig` on line 1 and `4` on line 2. Predict the number the program should print after `Coins in your purse:`.

Answer: .....

TASK 2 Two Bugs on the Road

EASY `a1.py` ~15 min

The starter's `east` and `west` hold one bug each, with no error shown. `east` prints nothing and leaves you at the tent. `west` at the tent prints nothing too, but it should print `Nothing lies west of Umro's tent.`

Your task: Find and fix the two bugs, one in `east` and one in `west`. Change nothing else.

Input: as in Task 1.

Output: the lines of Task 1, as before. Then two lines for each place you arrive at, its name and what is there, and the given lines.

Sample 1 (the checker uses this one)

Input

```
Sara
9
east
west
west
quit
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 9
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
Nothing lies west of Umro's tent.
You give up the quest.
```

Why: The tent to the bazaar, and back. The second `west` finds no road west, so you stay.

Hint: print probes: `print("place is", place)` as the last line of each branch. Run Sample 1, read the numbers, delete the probes. `look`, which works already, says where the game thinks you are.

Pen on this sheet, Shell closed, before you write Task 2 in `a1.py`. Only calculators are allowed.

Key 2.1 Predict what this code prints.

```
1 gold = 4
2 gold + 3
3 print(gold)
```

Answer: .....

Key 2.2 Predict what this code prints.

```
floor = 0
if floor >= 0:
    floor = floor - 1
print(floor)
```

Answer: .....

TASK 3 The Purse

EASY a1.py ~10 min

The purse holds the coins you carry.

Your task: Add the `purse` command: print the coins, or say that the purse is empty.

Input: as in Task 1.

Output: the lines of Tasks 1 and 2, as before. For `purse`, one line: `Coins in your purse: C` if C, the coins, is more than 0, or `Your purse is empty.` if C is 0.

Sample 1 (the checker uses this one)

Input

```
Sara
9
purse
quit
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 9
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
Coins in your purse: 9
You give up the quest.
```

Why: 9 is more than 0, so `purse` prints the number.

Hint: an `elif` branch at the Task 3 comment, with an `if/else` inside.

Pen on this sheet, Shell closed, before you write Task 3 in `a1.py`. Only calculators are allowed.

Key 3.1 The user inputs `Omar`, `4`, `east`, `purse`, `quit`. Predict how many lines the program should print.

Answer: .....

Key 3.2 The user inputs `Omar`, `5`, then `Purse` with a capital P. Which line should the program print for it?

(a) `Coins in your purse: 5`
(b) `I don't understand that.`
(c) Nothing at all.

TASK 4 Counting Steps

EASY a1.py ~10 min

⌨ Every command you type is one **step**, even one the game does not understand. `quit` is not a step.

Your task: Count the steps in `steps`. Add the `steps` command: it prints the count so far, itself included.

Input: as in Task 1.

Output: the lines of Tasks 1 to 3, as before. For `steps`, one line: `Steps so far: S`.

Sample 1 (the checker uses this one)

Input

```
Sara
9
east
dance
steps
quit
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 9
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
I don't understand that.
Steps so far: 3
You give up the quest.
```

Why: `east`, `dance` and `steps` make three steps. `dance` is not a command, but it counts.

Hint: the Task 4 comment marks the place for the counting line.

Pen on this sheet, Shell closed, before you write Task 4 in `a1.py`. Only calculators are allowed.

Key 4.1 The user inputs `Omar`, `3`, `east`, `purse`, `west`, `steps`, `quit`. Predict the number the program should print after `Steps so far:`.

Answer: .....

Key 4.2 The user inputs `Omar`, `3`, `steps`, `steps`, `quit`. Predict the number the program should print after `Steps so far:` the second time.

Answer: .....

2 The Bazaar and the Tent (~65 min)

Each new command checks a few things in order and answers with the first that holds, so the order matters.

TASK 5 A Bag of Salt

MEDIUM a1.py ~30 min

⌨ Salt keeps a bhoot away. The merchant at the bazaar sells a bag for `SALT_PRICE` coins, and you need only one.

Your task: Add `get salt`. Check the place, then the bag, then the coins.

Input: as in Task 1.

Output: the lines of Tasks 1 to 4, as before. For `get salt`, one line, the first that fits:

1. `There is no salt here.` away from the bazaar.
2. `You already have salt.` if you bought it before.
3. `You cannot afford salt.` if the coins are fewer than `SALT_PRICE`.
4. `You buy a bag of salt for P coins.`, where P is `SALT_PRICE`. The coins go down by P.

Sample 1 (the checker uses this one)

Input

```
Sara
9
get salt
east
get salt
get salt
purse
quit
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 9
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
There is no salt here.
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
You buy a bag of salt for 2 coins.
You already have salt.
Coins in your purse: 7
You give up the quest.
```

Why: The tent has no salt. The first `get salt` costs 2, and the second finds salt already: $9 - 2 = 7$.

Hint: use `has_salt`, and one `if/elif/else` chain in the order of the lines.

Pen on this sheet, Shell closed, before you write Task 5 in `a1.py`. Only calculators are allowed.

Key 5.1 The user inputs `Omar`, `1`, `get salt`, `quit`. In the tent with 1 coin, two checks fail. Which line should the program print for `get salt`?

(a) `There is no salt here.`
(b) `You cannot afford salt.`
(c) Both lines, in that order.

Key 5.2 The user inputs `Omar`, `2`, `east`, `get salt`, `purse`, `quit`. Which line should the program print for `purse`?

(a) `Your purse is empty.`
(b) `Coins in your purse: 2`
(c) `Coins in your purse: 0`

TASK 6 The Jinn's Bottle

MEDIUM a1.py ~30 min

A jinn lives in a brass bottle in Umro's tent. Each rub is a wish of `WISH_COINS` coins, and there are only `WISHES` wishes.

Your task: Add `rub`. In the tent, grant a wish while any are left.

Input: as in Task 1.

Output: the lines of Tasks 1 to 5, as before. For `rub`, one line, the first that fits:
1. `There is nothing here to rub.` away from the tent.
2. `The bottle is empty.` if no wishes are left.
3. `A jinn grants you W coins. Wishes left: L`, where W is `WISH_COINS` and L is the wishes left after this one. The coins go up by W.

Sample 1 (the checker uses this one)

Input

```
Sara
0
rub
rub
east
rub
purse
quit
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 0
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
A jinn grants you 5 coins. Wishes left: 2
A jinn grants you 5 coins. Wishes left: 1
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
There is nothing here to rub.
Coins in your purse: 10
You give up the quest.
```

Why: Two wishes of 5 coins turn 0 into 10. The bottle stays in the tent.

Hint: `wishes_left` starts at `WISHES`. Take 1 from it before you print.

Pen on this sheet, Shell closed, before you write Task 6 in `a1.py`. Only calculators are allowed.

Key 6.1 The user inputs `Omar`, `0`, then `rub` four times, then `quit`. Predict how many lines starting with `A jinn` the program should print.

Answer: .....

Key 6.2 Suppose the `ROAD` block said `WISHES = 1` and `WISH_COINS = 4`. The user inputs `Omar`, `3`, `rub`, `rub`, `purse`, `quit`. Predict the number the program should print after `Coins in your purse:` for `purse`.

Answer: .....

3 The Orchard, the Bridge and the Nest (~145 min)

Two creatures stand between you and the feather: the bhoot acts on its own, the dev only when you go east or pay.

TASK 7 The Bhoot

MEDIUM a1.py ~35 min

⌨ On every turn you stand in the orchard, the bhoot acts before you type, even when you type `quit`. It fears salt; else it snatches a coin.

Your task: Write the bhoot's rule at the starter's Task 7 comment: after the arrival lines, before the command is read.

Input: as in Task 1.

Output: the lines of Tasks 1 to 6, as before. On each turn in the orchard, one more line, the first that fits:

1. `The bhoot smells your salt and keeps away.` if you have salt.
2. `The bhoot snatches a coin!` if you have a coin. One coin goes.
3. `The bhoot wails at your empty purse.` otherwise.

Sample 1 (the checker uses this one)

Input

```
Sara
1
east
east
purse
quit
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 1
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
== The Haunted Orchard ==
A bhoot, a ghost, guards a pomegranate tree. It fears salt.
The bhoot snatches a coin!
Your purse is empty.
The bhoot wails at your empty purse.
You give up the quest.
```

Why: Two turns start in the orchard: before `purse` (1 coin) and before `quit` (none).

Hint: a rule, not a command: it stays outside the chain of commands.

Pen on this sheet, Shell closed, before you write Task 7 in `a1.py`. Only calculators are allowed.

Key 7.1 The user inputs `Omar`, `6`, then `east` three times, then `purse`, `quit`. Predict the number the program should print after `Coins in your purse:` for `purse`.

Answer: .....

Key 7.2 The user inputs `Omar`, `6`, `east`, `east`, `purse`, `west`, `purse`, `quit`. Predict the number the program should print after `Coins in your purse:` the last time.

Answer: .....

Key 7.3 The user inputs `Omar`, `3`, `east`, `get salt`, `east`, `purse`, `purse`, `quit`. Predict how many times the program should print `The bhoot smells your salt and keeps away.`

Answer: .....

TASK 8 The Pomegranate

MEDIUM a1.py ~25 min

⌨ One pomegranate tree grows in the orchard. The bhoot guards it from anyone who carries no salt, and you can carry only one pomegranate.

Your task: Add `get pomegranate`. Check the place, then the salt, then what you carry.

Input: as in Task 1.

Output: the lines of Tasks 1 to 7, as before. For `get pomegranate`, one line, the first that fits:

1. `There is no pomegranate here.` away from the orchard.
2. `The bhoot guards the tree.` if you have no salt.
3. `You already have a pomegranate.` if you carry one.
4. `You pick a pomegranate.` Now you carry one.

Sample 1 (the checker uses this one)

Input

```
Sara
5
east
east
get pomegranate
west
get salt
east
get pomegranate
quit
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 5
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
== The Haunted Orchard ==
A bhoot, a ghost, guards a pomegranate tree. It fears salt.
The bhoot snatches a coin!
The bhoot guards the tree.
The bhoot snatches a coin!
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
You buy a bag of salt for 2 coins.
== The Haunted Orchard ==
A bhoot, a ghost, guards a pomegranate tree. It fears salt.
The bhoot smells your salt and keeps away.
You pick a pomegranate.
The bhoot smells your salt and keeps away.
You give up the quest.
```

Why: Without salt the bhoot guards the tree, and takes a coin on each of Sara's two turns there. With salt, the pomegranate is hers.

Hint: use `has_fruit`. `get pomegranate` has the shape of `get salt`.

Pen on this sheet, Shell closed, before you write Task 8 in `a1.py`. Only calculators are allowed.

Key 8.1 The user inputs `Omar`, `3`, `east`, `get pomegranate`, `quit`. At the bazaar without salt, two checks fail. Which should the program print for `get pomegranate`?

(a) `The bhoot guards the tree.`
(b) Both lines, the bhoot's first.
(c) `There is no pomegranate here.`

Key 8.2 The user inputs `Omar`, `4`, `east`, `get salt`, `east`, `get pomegranate`, `get pomegranate`, `quit`. Predict how many times the program should print `You pick a pomegranate.`

Answer: .....

TASK 9 The Dev's *(Giant's)* Toll

MEDIUM a1.py ~40 min

The dev on the bridge wants one pomegranate as his toll, once. Coins do not move him. Until paid, he blocks the way east, but never the way west.

Your task: Add `pay`, and make `east` stop at the bridge until the toll is paid.

Input: as in Task 1.

Output: the lines of Tasks 1 to 8, as before. For `pay`, one line, the first that fits:

1. `There is nobody here to pay.` away from the bridge.
2. `The dev waves you across.` if you paid before.
3. `The dev laughs. "No pomegranate, no bridge."` if you carry no pomegranate.
4. `You give the dev your pomegranate. The dev steps aside.` Now you carry none.

For `east` on the bridge before you pay: `The dev blocks the bridge. "Pay first."`

Sample 1 (the checker uses this one)

Input

```
Sara
5
east
get salt
east
get pomegranate
east
east
pay
east
quit
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 5
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
You buy a bag of salt for 2 coins.
== The Haunted Orchard ==
A bhoot, a ghost, guards a pomegranate tree. It fears salt.
The bhoot smells your salt and keeps away.
You pick a pomegranate.
The bhoot smells your salt and keeps away.
== The Dev's Bridge ==
A dev, a giant, wants a pomegranate to let you pass.
The dev blocks the bridge. "Pay first."
You give the dev your pomegranate. The dev steps aside.
== The Simurgh's Nest ==
A golden feather lies in the nest.
You give up the quest.
```

Why: Salt first, then the pomegranate. The fourth `east` is blocked. After the toll, `east` reaches the nest.

Hint: `pay` has the shape of `get salt`; for `east`, one `elif` at the Task 9 comment.

Pen on this sheet, Shell closed, before you write Task 9 in `a1.py`. Only calculators are allowed.

Key 9.1 The user inputs `Omar`, `100`, then `east` three times, then `pay`, `quit`. Which line should the program print for `pay`?

(a) `You give the dev your pomegranate. The dev steps aside.`
(b) `There is nobody here to pay.`
(c) `The dev laughs. "No pomegranate, no bridge."`

Key 9.2 The user inputs `Omar`, `5`, `east`, `get salt`, `east`, `get pomegranate`, `east`, `pay`, `west`, `get pomegranate`, `get pomegranate`, `quit`. Predict how many times the program should print `You pick a pomegranate.`

Answer: .....

Key 9.3 Omar stands on the bridge and has not paid. The user inputs `west`. Which line should the program print next?

(a) `The dev blocks the bridge. "Pay first."`
(b) `== The Haunted Orchard ==`
(c) `Nothing lies west of Umro's tent.`

TASK 10 The Feather

MEDIUM a1.py ~40 min

⌨ Taking the feather in the nest wins at once. Umro judges you by your steps: 10 or fewer is a true trickster, 15 or fewer a good apprentice.

Your task: Add `get feather`. At the nest it wins and ends the loop. After the loop, print a winner's verdict.

Input: as in Task 1, but a game may end with `get feather` in place of `quit`.

Output: the lines of Tasks 1 to 9, as before. For `get feather`: `There is no feather here.` away from the nest, and at the nest `You take a golden feather from the nest.`, and the game ends. After the loop, for a winner only:
Line 1: `Umro Ayyar smiles. Steps: S`, where S is the steps.
Line 2: `A true trickster!` if S is 10 or fewer, `A good apprentice.` if 15 or fewer, else `Slow, but the feather is home.`

Sample 1 (the checker uses this one)

Input

```
Sara
6
east
get salt
east
get pomegranate
east
pay
east
get feather
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 6
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
You buy a bag of salt for 2 coins.
== The Haunted Orchard ==
A bhoot, a ghost, guards a pomegranate tree. It fears salt.
The bhoot smells your salt and keeps away.
You pick a pomegranate.
The bhoot smells your salt and keeps away.
== The Dev's Bridge ==
A dev, a giant, wants a pomegranate to let you pass.
You give the dev your pomegranate. The dev steps aside.
== The Simurgh's Nest ==
A golden feather lies in the nest.
You take a golden feather from the nest.
Umro Ayyar smiles. Steps: 8
A true trickster!
```

Why: Eight commands, `get feather` among them, make 8 steps: 10 or fewer.
Hint: set `won` to `True`, then `break`. After the loop, `if won:` keeps the verdict from a quitter.

Pen on this sheet, Shell closed, before you write Task 10 in `a1.py`. Only calculators are allowed.

Key 10.1 The user inputs `Omar`, `8`, `east`, `get salt`, `east`, `get pomegranate`, `east`, `pay`, `east`, `get feather`. Predict the number the program should print after `Steps:`.

Answer: .....

Key 10.2 A winner's game counts exactly 15 steps. Which verdict line should the program print?
(a) `A true trickster!`
(b) `A good apprentice.`
(c) `Slow, but the feather is home.`

Key 10.3 The user inputs `Omar`, `9`, `east`, `get salt`, `east`, `get pomegranate`, `east`, `pay`, `east`, `quit`. Predict how many lines the program should print after `== The Simurgh's Nest ==`.

Answer: .....

4 Bonus: For the Brave (~60 min)

For students who finish early and want more: Task 11 is worth 20 marks on top of the 100.

TASK 11 **Umro's Map** HARD a1.py ~60 min

A map shows the road in one line, a cell a place. `[@]` is where you stand, `[o]` a place reached before, `[?]` one not reached yet.

Your task: Keep the farthest place reached in `farthest`. Add `map`: build the line with a `for` loop, then print it.

Input: as in Task 1.

Output: the lines of Tasks 1 to 10, as before. For `map`, one line: `Map:`, a space, and five cells, from `TENT` to `NEST`.

Sample 1 (the checker uses this one)

Input

```
Sara
5
map
east
east
west
map
quit
```

Output

```
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 5
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
Map: [@][?][?][?][?]
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
== The Haunted Orchard ==
A bhoot, a ghost, guards a pomegranate tree. It fears salt.
The bhoot snatches a coin!
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
Map: [o][@][o][?][?]
You give up the quest.
```

Why: Sara reached the orchard and came back to the bazaar. The tent and the orchard are reached.
Hint: only `east` reaches a new place. Start the line as `"Map: "` and add a cell each time round.

Pen on this sheet, Shell closed, before you write Task 11 in `a1.py`. Only calculators are allowed.

Key 11.1 The user inputs `Omar`, `5`, then `east` three times, `west` twice, then `map`, `quit`. Predict how many `[o]` cells the map line should show.

Answer: .....

Key 11.2 The program printed `Map: [o][o][@][o][?]`. Counting the tent as 0, work out the number of the farthest place reached.

Answer: .....

Marking and Hand-in

Marks. Tasks 1 to 10 are worth 10 each: 40 for Tasks 1 to 4 (EASY), 60 for Tasks 5 to 10 (MEDIUM). Task 11 (HARD, bonus): 20, on top of the 100. `check.py`'s points are a first mark only. After the deadline, each task is replayed on three to five hidden runs, with other names, coins, commands and `ROAD` numbers. Full marks when all its runs pass, half when only its sample passes; half, too, for a tool beyond Lecture 8.

Before you hand in

- ☐ `check.py` says PASS for Tasks 1 to 10 (and 11, if you did it), and you played the game from the tent to the feather.
- ☐ Your code uses the `ROAD` names, never their numbers, and the block is as the starter gave it.
- ☐ Every `input()` is empty, and no print probe is left.
- ☐ No `def`, lists, `import`, or anything with a dot.
- ☐ Every block of code has a comment that says what it does.
- ☐ You typed every line yourself, and can explain every line.

Appendix: A Winning Run

One whole game, from Umro's tent to the feather, played by the finished `a1.py`. This is how it looks in the Shell: the blue lines are what Sara types, and the rest is what the game prints. Your `a1.py` must print the same lines for the same input. The `map` line needs Task 11, the bonus; without it, `map` prints `I don't understand that.`

```
Sara
3
Umro Ayyar: "Sara, bring me a Simurgh feather."
Coins in your purse: 3
== Umro's Tent ==
A brass bottle stands here. A jinn lives in it.
purse
Coins in your purse: 3
rub
A jinn grants you 5 coins. Wishes left: 2
east
== The Bazaar ==
A merchant sells salt: 2 coins a bag.
get salt
You buy a bag of salt for 2 coins.
east
== The Haunted Orchard ==
A bhoot, a ghost, guards a pomegranate tree. It fears salt.
The bhoot smells your salt and keeps away.
get pomegranate
You pick a pomegranate.
The bhoot smells your salt and keeps away.
east
== The Dev's Bridge ==
A dev, a giant, wants a pomegranate to let you pass.
east
The dev blocks the bridge. "Pay first."
pay
You give the dev your pomegranate. The dev steps aside.
map
Map: [o][o][o][@][?]
east
== The Simurgh's Nest ==
A golden feather lies in the nest.
look
== The Simurgh's Nest ==
A golden feather lies in the nest.
steps
Steps so far: 13
get feather
You take a golden feather from the nest.
Umro Ayyar smiles. Steps: 14
A good apprentice.
```

Why: Sara types 14 commands after the name and the coins, and each one is a step, `look` too. 14 is more than 10 but 15 or fewer: a good apprentice.