Family edition

Telescope Lab: The Night's Queue, a Greedy Scheduler

Ages 14-18 · about 45 minutes, in one sitting

In Telescope Lab your child schedules a mountaintop telescope's night in Python, and writes a greedy algorithm that turns 6 points into 23.

What you need

- A computer or Chromebook with a current browser.
- The link: mytekdev.com/try/telescopelab?lesson=the-nights-queue-a-greedy-scheduler&ref=fam-telescopelab-4. No account, no email and nothing to install.

Before you start: try it once yourself

Open mytekdev.com/try/telescopelab?lesson=the-nights-queue-a-greedy-scheduler&ref=fam-telescopelab-4. A short guide opens over Level 8, The night's queue, with the program in the Python tab. The chart shows each target's height through the night with the 30 degree line, and the log under it lists every block and every try that could not work. The night waits for the program: nothing moves until you press play. Speed up three times (1000 times faster) runs the 10 hour night in under a minute.

1. **The targets.** M92 (priority 5, 60 minutes), M104 the Sombrero (4, 60), M51 the Whirlpool (3, 60), M81 (3, 60), M66 (2, 45), Jupiter (2, 30) and Uranus (4, 60): 23 points. A block counts only if the target stays above 30 degrees in the dark for all of it; a try that cannot work costs 10 minutes.
2. **The starter.** For each target in the list's order, observe it. M92, M104, M51, M81 and M66 have not risen high enough (7:35 to 8:25 pm, 10 minutes each), then Jupiter (8:25 to 8:55) and Uranus (8:55 to 9:55) count. The program ends at 9:55 pm with 6 of 23.
3. **The scheduler.** Uranus at 7:35 pm, Jupiter 8:35, M81 9:05, M66 10:05, then waits until M51 is high enough at 11:40 pm, M104 at 12:40 am (18 points at 1:40 am, the level done), waits again, and M92 at 3:30 am: 23 by dawn, no failed tries.
4. **The versions.** (1) No check: it tries M92, the highest priority, every 10 minutes from 7:35 pm until it is high enough at 3:25 am (47 failed tries), then M92 and M104, 9 points; one more failed try on Uranus as dawn comes (48 in all), and dawn stops the program ("Dawn: the night is over"). (2) No priority comparison: 23, the same plan. (3) wait 1800: 23, M104 at 11:50 pm before M51, 18 at 1:50 am. (4) One for loop with the check: Uranus only, 4 points, and the program ends. With wait 60: 18 at 1:36 am. None of these change from run to run.

The program. Indent is how many levels in (four spaces each):

INDENT	LINE
0	from telescope import Telescope

INDENT	LINE
0	scope = Telescope()
0	while True:
1	best = None
1	for t in scope.targets():
2	if not scope.done(t) and scope.minutes_left(t) >= scope.needs(t):
3	if best is None or scope.priority(t) > scope.priority(best):
4	best = t
1	if best is None:
2	scope.wait(600)
1	else:
2	scope.observe(best)

> This is the program on the worksheet. A child who types it with the indents as shown reaches 23.

The ideas, in plain words

Real observatories plan every night: which targets, in what order, given when each is high enough, how dark it is, where the Moon is, and how much each program counts. A greedy scheduler, at each free moment the best target that can be done now, is simple and fast, and real schedulers start from it and add tweaks. It is not always best for the whole night: a greedy choice now can use time that a better plan would save for something that rises later.

Tonight's seven targets fit in the night with room to spare, as long as no time is wasted on tries that cannot work and the program comes back for targets that rise later. That is why the check and the wait do the work and the priority comparison changes only the order. The summit, the targets' heights (from Astronomy Engine) and the 30 degree limit are the app's model of a real 10 meter class telescope's night; the targets' positions come from OpenNGC.

Astronomers, telescope operators (the job this app is closest to), engineers who build cameras and adaptive optics, and data scientists who sort survey pictures all work with telescopes like these. Anyone can help with real data through citizen science projects such as Zooniverse and the AAVSO.

Do it together

1. Open mytekdev.com/try/telescopelab?lesson=the-nights-queue-a-greedy-scheduler&ref=fam-telescopelab-4, press `Speed up` three times, then play. Read the log under the chart.

   > **You'll know it worked when:** Five tries could not work (the targets had not risen), then Jupiter and Uranus: 6 of 23, and the program ends at 9:55 pm.

2. Open `Python` and type the scheduler printed in this plan (a table of lines, with how far each is indented) in place of the starter. Go back to `Simulator` and play.

   > **If you get stuck:** Check the indentation: `if best is None:` lines up with `for t in`.
   >
   > **You'll know it worked when:** 18 points at 1:40 am, and 23 of 23 by dawn, with no failed tries.

3. Take out `and scope.minutes_left(t) >= scope.needs(t)` and play.

   > **You'll know it worked when:** It tries M92, the highest priority, over and over until it rises; 9 points, and the program stops at dawn.

4. Put it back, and change `if best is None or scope.priority(t) > scope.priority(best):` to `if best is None:`. Play.

   > **You'll know it worked when:** Still 23: tonight there is time for everything, so the order of choosing did not matter.

5. Change `scope.wait(600)` to `scope.wait(1800)` and play. Compare the order with the first run.

   > **You'll know it worked when:** M104 now comes before M51: both are up when it looks at 11:50 pm, and priority picks M104.

Talk about it

Ask these while you work together. A child who has understood it can answer them without help:

- Why did the starter score only 6?
- What does scope.minutes_left(t) >= scope.needs(t) check, and why does the scheduler need it?
- Why does the scheduler need a wait, and what would happen without one?
- Why did removing the priority comparison change nothing tonight?
- Why is a while loop the right structure here and a for loop not?

> The fifth question is the one that matters. A child who says "a for loop goes through the list once, but targets rise later, so the program has to keep choosing until the night ends" can justify a control structure. A child who says "the while loop worked" has the result but not the idea.

For younger and older kids

Younger: For a younger child, run the starter and the scheduler, and talk through the log: what did each one try, and when?

Older: For an older child, write the choice as a function, `best_target()`, and try preferring the target that sets soonest.

Keep going

- Make the choice smarter: among the targets that can be done now, prefer the one that sets soonest. Does it change the order or the time 18 is reached?
- Count how many times the program asked scope.minutes_left in the whole night. How could it ask less?
- Write a version that plans the whole night first, from the heights at each hour, then observes in that order. Is it better than greedy here?

Bring it into other subjects

- Math: from the chart, estimate when each target rises above 30 degrees. Check your estimates against scope.minutes_left at those times.
- Earth science: why do the galaxies rise later in the night than Uranus sets? Use the idea that the sky turns 15 degrees an hour.
- Life skills: plan a Saturday with errands that are open only at certain times, greedily, then by planning ahead. Which is better?
- Careers: look up what a telescope scheduler or night assistant does at a large observatory.

For your records

Ready to copy into an hour log, a portfolio or a transcript.

LESSON	Telescope Lab: The Night's Queue, a Greedy Scheduler
SUBJECTS	Computer science
TIME	45 minutes
WHAT WE DID	Wrote a greedy scheduler in Python for a simulated observatory's night, raised the score from 6 to 23 of 23 points, and tested which parts of the rule mattered by removing them one at a time.
SKILLS PRACTICED	Greedy algorithms; Loops and selection; Testing a program

Activity sheet (optional)

Name: _____ Date: _____

Open the app: scan the QR code, or go to mytekdev.com/try/telescopelab?lesson=the-nights-queue-a-greedy-scheduler

Essential question: How does a program choose well, one step at a time?

Warm-up: Seven things to do in one evening, some only before 9 and some only after midnight: is the list's order a good plan? _____

The program. Indent is how many levels in (four spaces each):

INDENT	LINE
0	from telescope import Telescope
0	scope = Telescope()
0	while True:
1	best = None
1	for t in scope.targets():
2	if not scope.done(t) and scope.minutes_left(t) >= scope.needs(t):
3	if best is None or scope.priority(t) > scope.priority(best):
4	best = t
1	if best is None:
2	scope.wait(600)
1	else:
2	scope.observe(best)

1. **Vocabulary.** Greedy algorithm, priority, constraint, altitude, block, selection: write each in your own words.

   _____

   _____

   _____

   _____

2. **The starter.** Points: _____ of 23. Tries that could not work: _____. The program ended at _____.
3. **The scheduler.** Type the program above in place of the starter (its first two lines are the same). 18 points at _____. Points by dawn: _____. Failed tries: _____.

4. **Versions.** (1) No check: points _____ failed tries _____. (2) No priority comparison: _____ _____. (3) wait 1800: _____ _____, first after M66: _____. (4) One for loop with the check: _____ _____.

1. Which part of the rule did the work tonight, and how do your versions show it?

2. Why does the scheduler use a while loop, not a for loop over the targets?

Quick check

Name: _____

1. A greedy algorithm: (A) tries every possible order (B) makes the best choice it can at each step (C) takes the items in the order given

2. The starter scored 6 mainly because: (A) the targets were worth little (B) it tried targets before they rose and never came back (C) the night was too short

3. A friend's scheduler picks the highest priority target that is not done, with no other check. Use what you learned today to say what will happen, and what to add.

Answers

QUESTION	ANSWER	WHY
1	B	At each free moment it picks the best target that can be done now.
2	B	Five targets had not risen; each failed try cost 10 minutes and the for loop went through the list once.
3	Open answer	It keeps trying M92 before it rises, wasting 10 minutes each time (47 tries on M92, 9 points by dawn). Add the check that the target stays above 30 degrees for the minutes it needs, and wait when nothing can be done.

If your state or umbrella school asks, this lesson covers CSTA 2017 3A-AP-13, CSTA 2017 3A-AP-15, CSTA 2017 3B-AP-10.

Terms of use

This lesson was made by mytekDev. You may print and copy it for the children you teach at home, and change it for them. You may not sell it, share it with other families, or post it publicly.

The app this lesson uses is free and runs in the browser, with nothing to install and no account needed. It is at mytekdev.com.

© 2026 mytekDev. All rights reserved.