Computer Science Team Week 15

Henry Gustafson   Julian Bauer

The College Preparatory School

Computer Science Team
April 8, 2024

ACSL 4

Topics are Graph Theory, Digital Electronics, Assembly Language.

Fun coding problems

Theme: Spring Break

Problem Break Breaks

Problem Break Breaks Why did the calendar go to therapy? Because it had too many breaks and needed to spring back into action! (credit to ChatGPT for humor)
Given a list of breaks, where each break is a two-integer tuple containing the break's start time and end time, and the integer *threshold*, output whether for all pairs of consecutive breaks the difference between the first breaks end time and the second breaks start time is ≥ threshold.

```python
def break_breaks(
    breaks: list[tuple[int, int]],
    threshold: int,
) -> bool
```

Problem Break Breaks

Example

```
assert break_breaks(
    [(0, 1), (3, 4)], 3
) == false
assert break_breaks(
    [(0, 1), (3, 4)], 2
) == true
```

Problem Issues

Problem Issues You are making an issues in science presentation to show your students how few days are left in the school year. You represent the remaining time with a bunch of evenly-sized squares, colored grey if the time they represent is in a break but colored yellow if not. The squares can represent as many days as you want, but each square must be the same size. Given a list of breaks (in the same format as the last problem) and the last day of the school year (inclusive), output the minimum number of total squares to represent the time remaining exactly, and the number of yellow squares. Assume today is 0.

```
def squares(
    breaks: list[tuple[int, int]],
    end: int,
) -> (int, int)
```

Problem Issues

Example

```
assert squares(
    [(0, 1), (4, 5)],
    5,
) == (3, 1)
```

Problem Hamiltonian

Problem Hamiltonian Josh is constructing wheels out of rubber balls and springs. However, some of the springs have recently broken (a ”spring break”), and Josh wants to know which wheels still work. Josh knows that a given pile of springs and balls is a functional wheel if the graph created by treating the rubber balls as nodes and the springs as directed edges is *Hamiltonian* (has a closed path (cycle) that visits each node exactly once, ending on the same node as it began). Given a square, 2D boolean array representing the connections between nodes (where each row represents that node’s connections to other nodes by index) output whether the graph is a functioning wheel.

```
def hamiltonian(
    node_connections: list[list[int]]
) -> (int, int)
```

Problem Hamiltonian

Example

```
assert hamiltonian(
    [
    [0, 1, 0, 0], # node 0, directed edge to node 1
    [0, 0, 1, 0], # node 1, directed edge to node 2
    [0, 0, 0, 1], # node 2, directed edge to node 3
    [1, 0, 0, 0], # node 3, directed edge to node 0
    ]
) == true
# node 0 -> node 1 -> node 2 -> node 3 -> node 1
```

The End

Questions? Comments? Remarks?
Considerations? Confusions?