Name: _____

Today's Date: _____

Today's Goals

- Explain what the *amortized* cost of a function or algorithm is.
- Explain when amortized cost is a useful metric.
- Describe an algorithm for which amortized cost is a useful metric.

Today's Question(s)

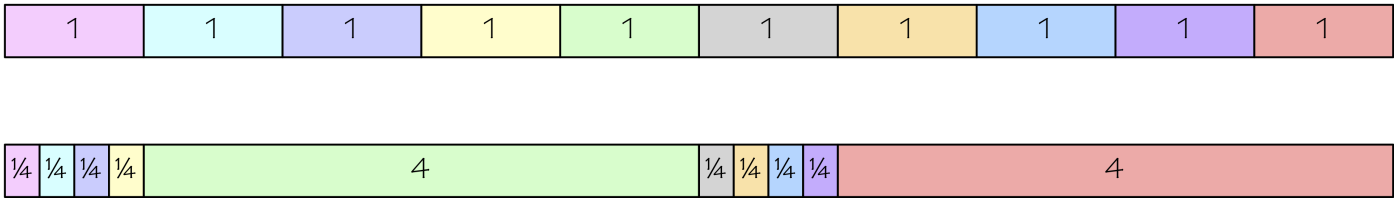
What is the *expected* cost of the following algorithm? You should use "number of rolls" as your cost metric.

```
while true:
    roll a die
    if the value is 6: break
```

Lingering Questions

Reminder: Time Complexity

Visualizing Time Usage

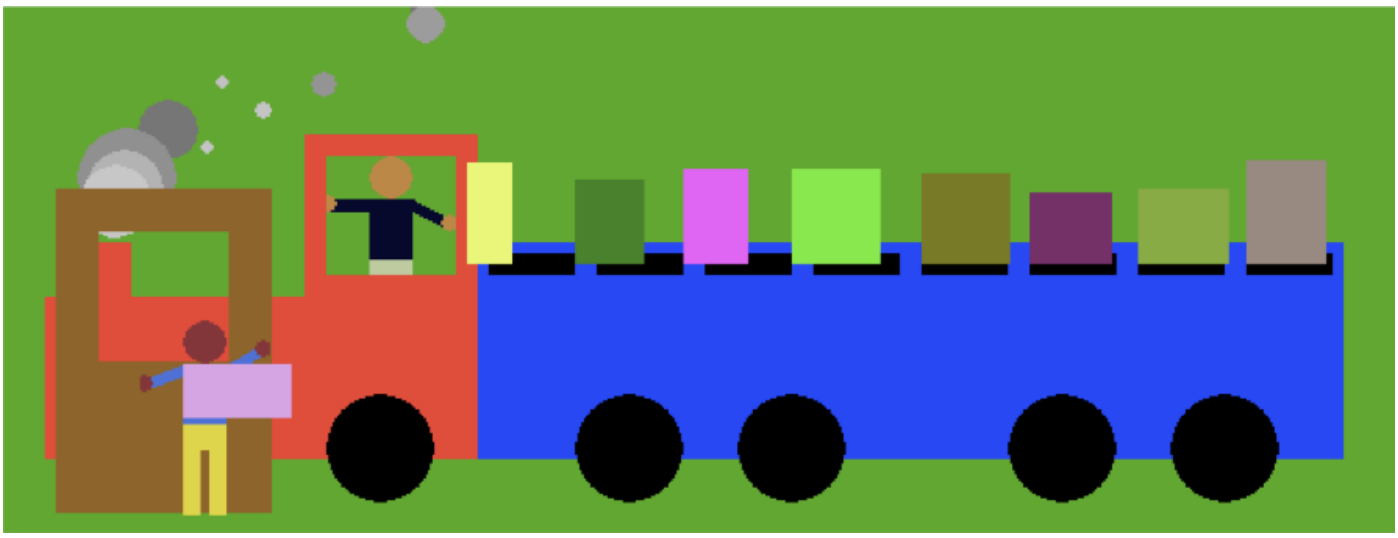


Bounding time for a sequence of operations

What's the worst case time for:

```
std::vector<int> v;for (int i = 0; i != m; ++i) {
    v.push_back(n);
}
```

Quinn's Story



Quinn's Problem: No idea how many packages will show up.

Scheduling Quinn's Time

We need to know how long it will take Quinn to load up all of the packages. We could:

- Let each customer wait as long as their package takes.
- Tell every customer to expect the longest possible loading time.
- Who will this work for? Who won't it work for?

Better Estimates

We can make a better estimate of the average time per customer by...

because ...

Class Exercise: Growable Arrays

Quinn's train doubled when it ran out of room.

- What else could we have done?
- How would that affect how much work Quinn had to do?

A Better Worst-Case Estimate

Let f be an operation we will carry out many times (like loading a new package)

Time to complete f for the ith time:

To perform f k times:

A "typical" instance will take:

Amortized cost is:

Banker's method

It's like insurance, because:

Bounding Time for a Sequence of Operations

```
std::vector<int> v;
for (int i = 0; i != m; ++i) {
    v.push_back(n);
}
```

What is the worst case time for...

- Just the last iteration?
- The entire loop?

Warning: Correctness

Summing amortized times only works when you start from "empty"

```
// Start with n empty strings
vector<string> ss{n};
// Let's do just a single push_back
ss.push_back("wow"); // Not worst-case O(1)!
```