Programming, Problem Solving, and Algorithms

CPSC203, 2019 W1

Announcements

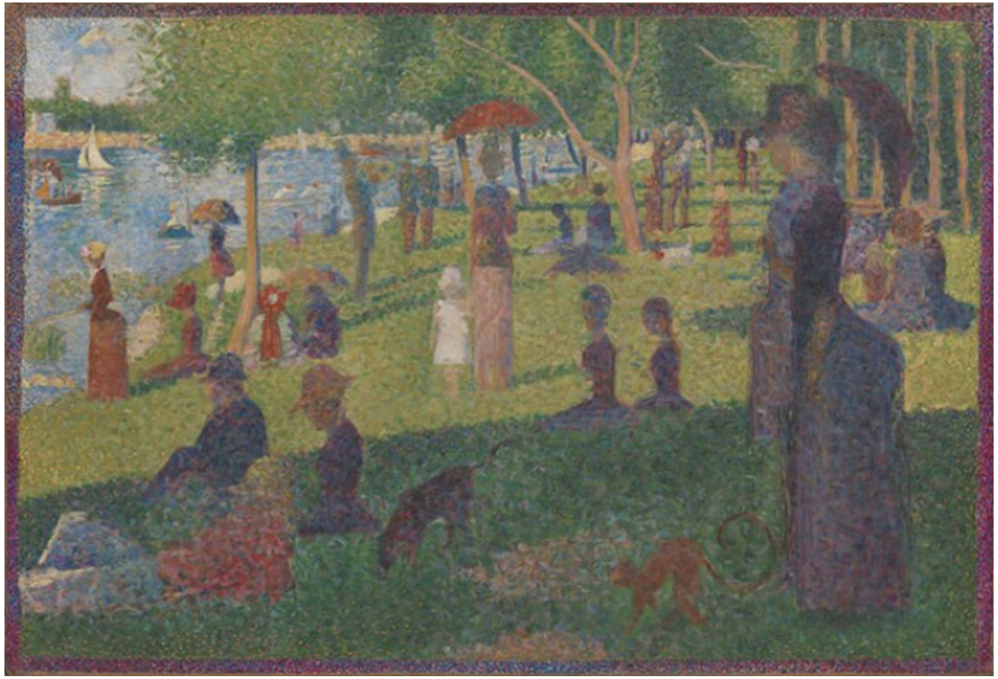
Project 1 is released. Due 11:59p, Oct 17.

“Problem of the Day” continues!

Today:

Introduction to BFS w application to Voronoi Art.

Pointillism

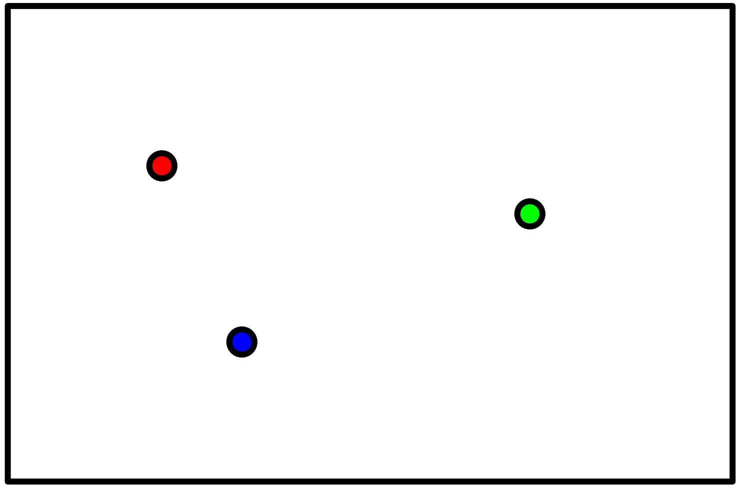

A Sunday on La Grande Jatte, Georges Seurat

Demo and Analysis OLD

https://github.students.cs.ubc.ca/cpsc203-2019w-t1/LecVor

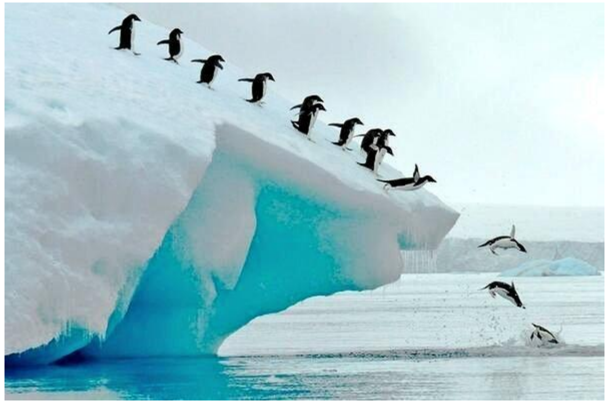
How much work is done?

1) Read image: $w * h$
2) Choose centers: $c = density * w * h$
3) Build new image: $c * w * h$
4) Write out new image: $w * h$



Data Structure: Queue

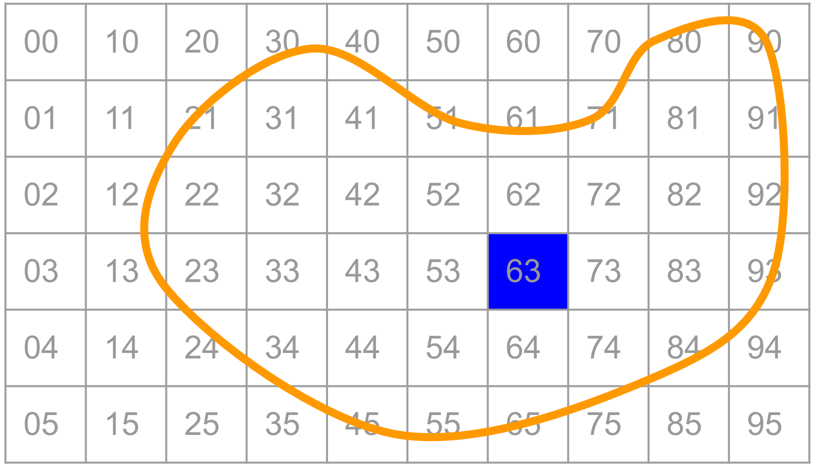
To orchestrate the fill, we’ll use a data structure called a QUEUE.



Queue:

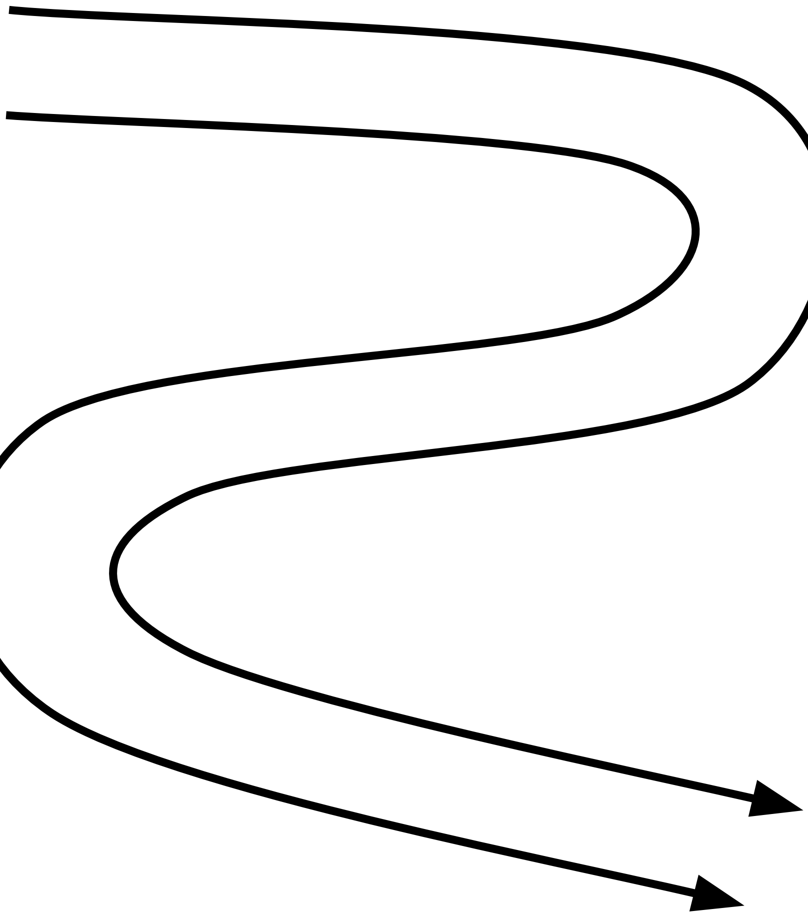
- enqueue(k) -- places data k onto the structure, at the “end”
- dequeue() -- removes and returns the “first” element from the structure

Putting it together



00	10	20	30	40	50	60	70	80	90
01	11	21	31	41	51	61	71	81	91
02	12	22	32	42	52	62	72	82	92
03	13	23	33	43	53	63	73	83	93
04	14	24	34	44	54	64	74	84	94
05	15	25	35	45	55	65	75	85	95

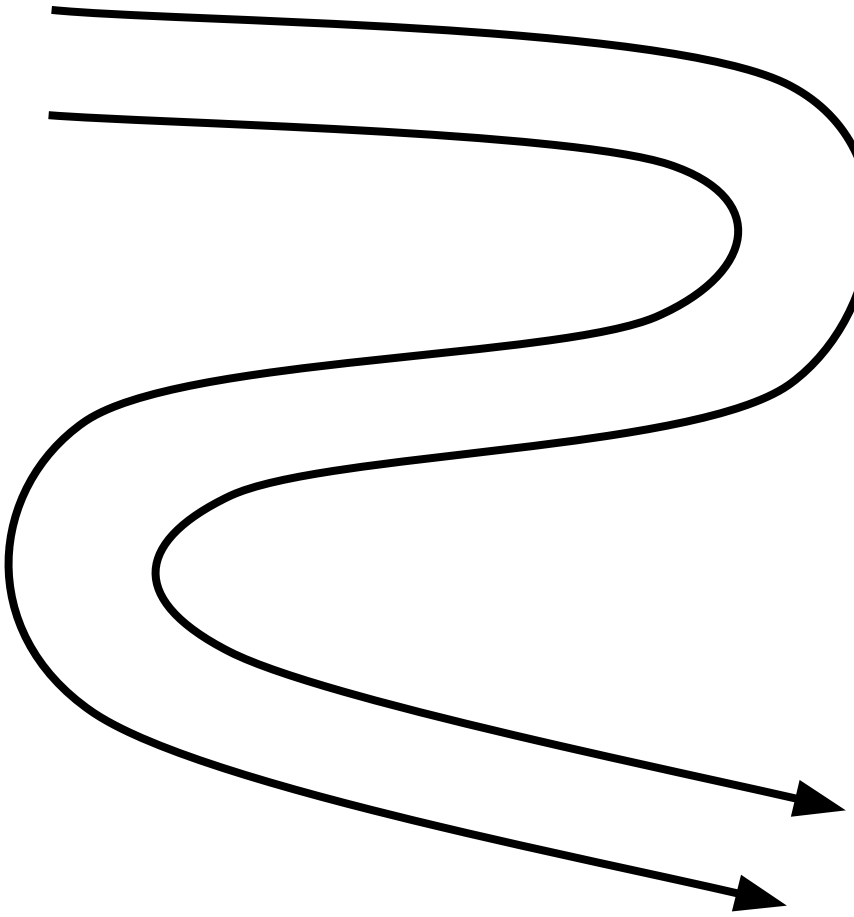
```
1)  enqueue the center to start
2)  while the queue is not empty:
      a)  v = dequeue
      b)  for each valid neighbor w, of v:
            i)  color w
           ii)  enqueue w
```



Putting it together

00	10	20	30	40	50	60	70	80	90
01	11	21	31	41	51	61	71	81	91
02	12	22	32	42	52	62	72	82	92
03	13	23	33	43	53	63	73	83	93
04	14	24	34	44	54	64	74	84	94
05	15	25	35	45	55	65	75	85	95

```
1)  enqueue the center   to start
2)  while the queue is not empty:
      a)  v = dequeue
      b)  for each valid neighbor w, of v:
            i)   color w
            ii)  enqueue w
```



Designing the solution

```
1)  enqueue the centers   to start
2)  while the queue is not empty:
    a)  v = dequeue
    b)  for each valid neighbor w, of v:
        i)  color w
        ii) enqueue w
```

1. What info should we put on the queue?

2. Remember we’re using `deque` as our queue (Python).

3. Do `deque`s have a way to check for empty?

4. What are the “neighbors” of pixel (x,y)?

5. What would be an *invalid* neighbor?
    -
    -

Demo and Analysis NEW

https://github.students.cs.ubc.ca/cpsc203-2019w-t1/LecBFS

How much work is done?

1) Read image: $w * h$

2) Choose centers: $c = density * w * h$

3) Build new image:

4) Write out new image: $w * h$

Graphs: A new model for representing images

00	10	20	30	40	50	60	70	80	90
01	11	21	31	41	51	61	71	81	91
02	12	22	32	42	52	62	72	82	92
03	13	23	33	43	53	63	73	83	93
04	14	24	34	44	54	64	74	84	94
05	15	25	35	45	55	65	75	85	95

A *Graph* is a collection of *vertices*, and *edges* between them. They're used as a general model for many problems.

In our images every __________ is a vertex, and every __________ is an edge. How many edges are there in the graph representing the image on the left?

Our fast algorithm for Voronoi Art mirrors a classic algorithm on graphs called Breadth First Search.

Breadth First Search

Breadth-first search (**BFS**) is an algorithm for traversing or searching tree or graph data structures. It starts at the tree root (or some arbitrary node of a graph, sometimes referred to as a 'search key'[1]), and explores all of the neighbor nodes at the present depth prior to moving on to the nodes at the next depth level. (--Wikipedia)

Simplified description:

POTD #16 Thu

https://github.students.cs.ubc.ca/cpsc203-2019w-t1/potd16

Describe any snags you run into:

1. Line ___: ______________________________
2. Line ___: ______________________________
3. Line ___: ______________________________
4. Line ___: ______________________________
5. Line ___: ______________________________

ToDo for next class...

POTD: Continue every weekday! Submit to repo.

Reading: TLACS Ch 10 & 12 (lists and dictionaries)

References:

https://en.wikipedia.org/wiki/Voronoi_diagram