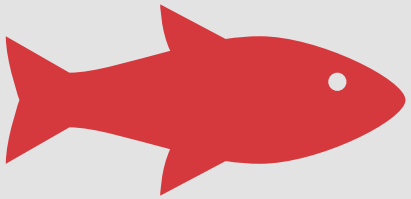
Genome Assembly for Biocontrol of Invasive Carp

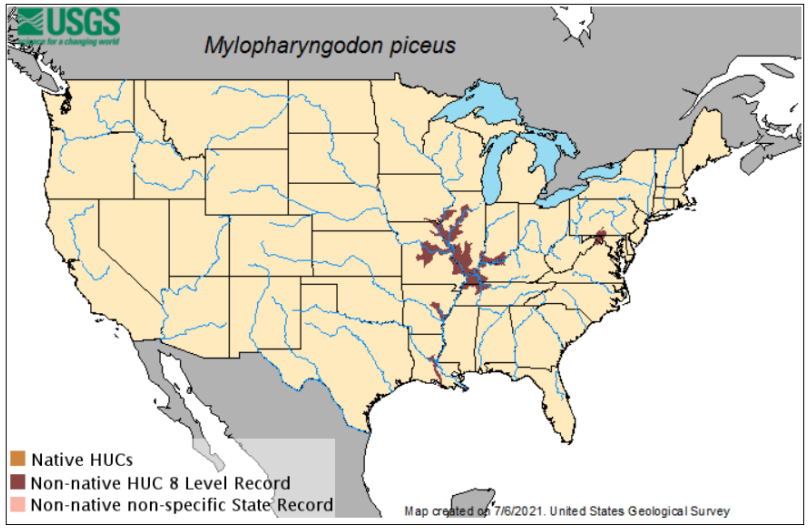
A DFP project brought to you by: Emma Schillerstrom



Black Carp

Mylopharyngodon piceus

- Native to East Asia
- Brought to U.S. in 1970s
- Prey on mussels and snails
  - Pros:
    - Utilized for aquaculture
  - Cons:
    - Increased their use & spread
    - Reduces water quality
    - Wipes out native populations



Current Genetic Biocontrol Techniques

SIT = sterile insect technique

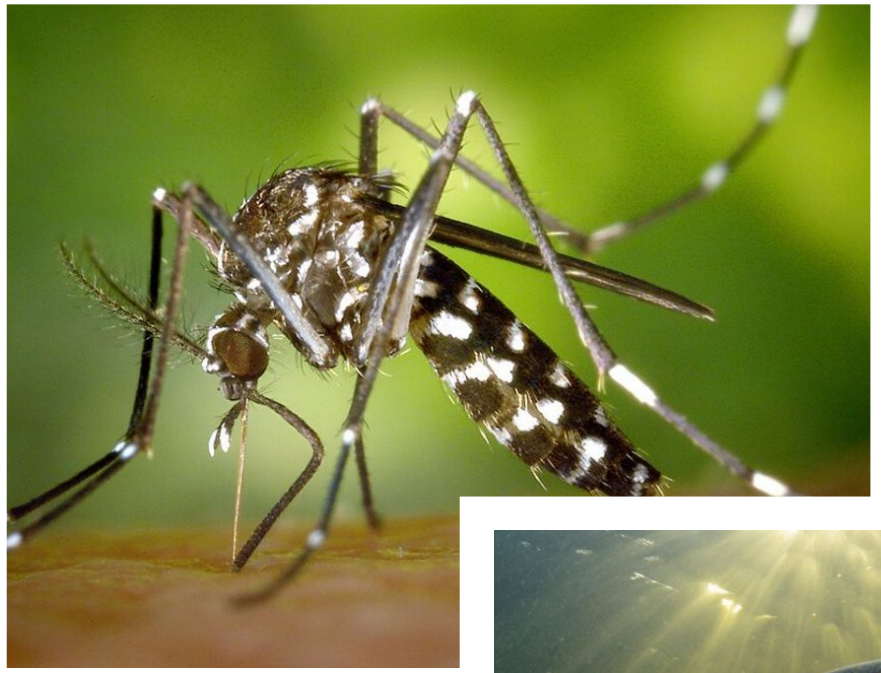
IIT = incompatible insect technique

RIDL = release of insects carrying dominant lethal

YY males = eliminating female sex

TFT = Trojan female technique

Gene drive = accelerates propagation of modified genes



Project Objective

Develop a pipeline to perform de novo genome assembly from Black Carp HiFi reads using Yeti.

Project Objective

chain of computing programs

from scratch

Develop a pipeline to perform de novo genome assembly from Black Carp HiFi reads using Yeti.

piecing together fragmented sequences into one big picture

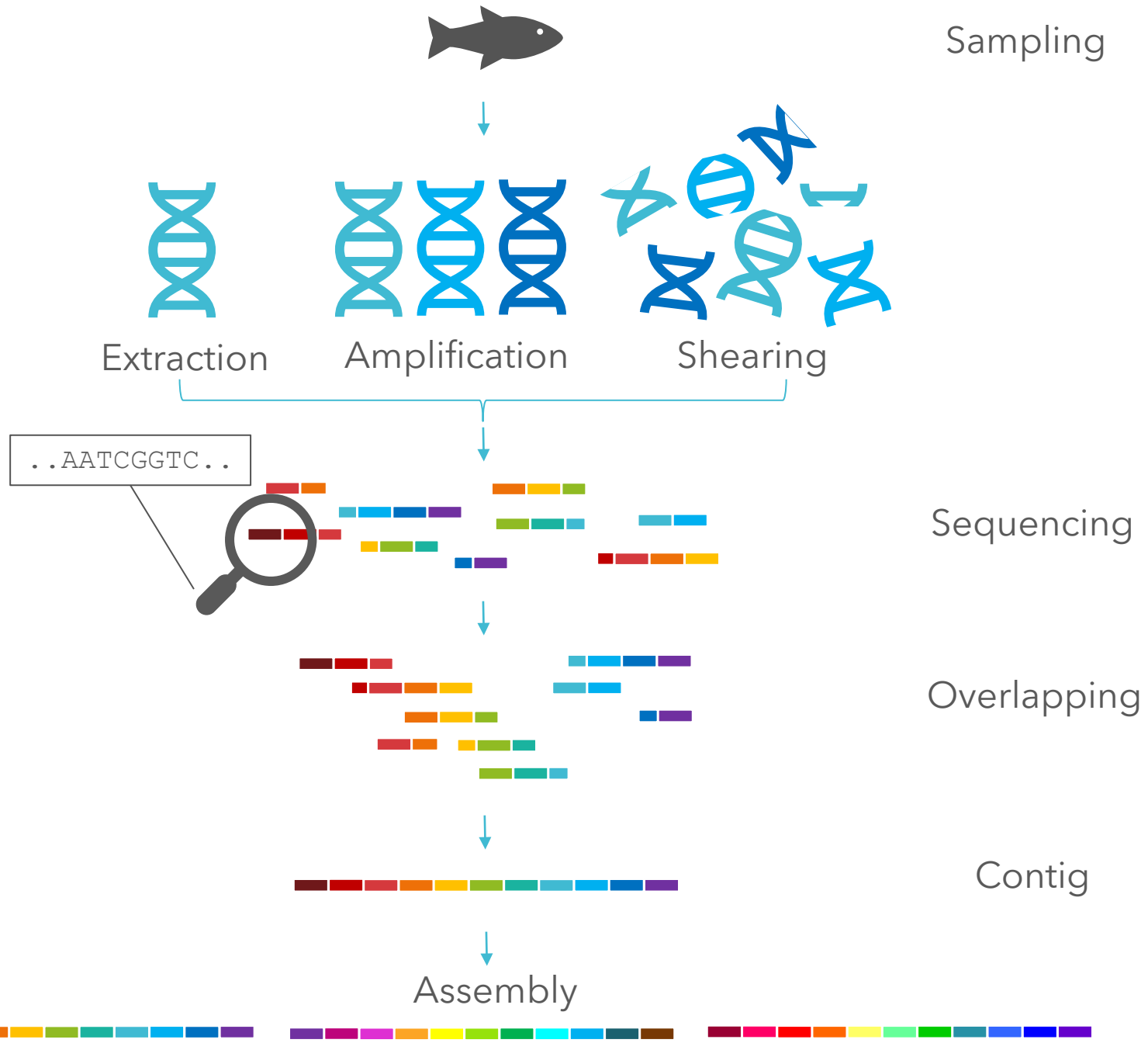
DNA sequences

supercomputer

What does de novo assembly look like?

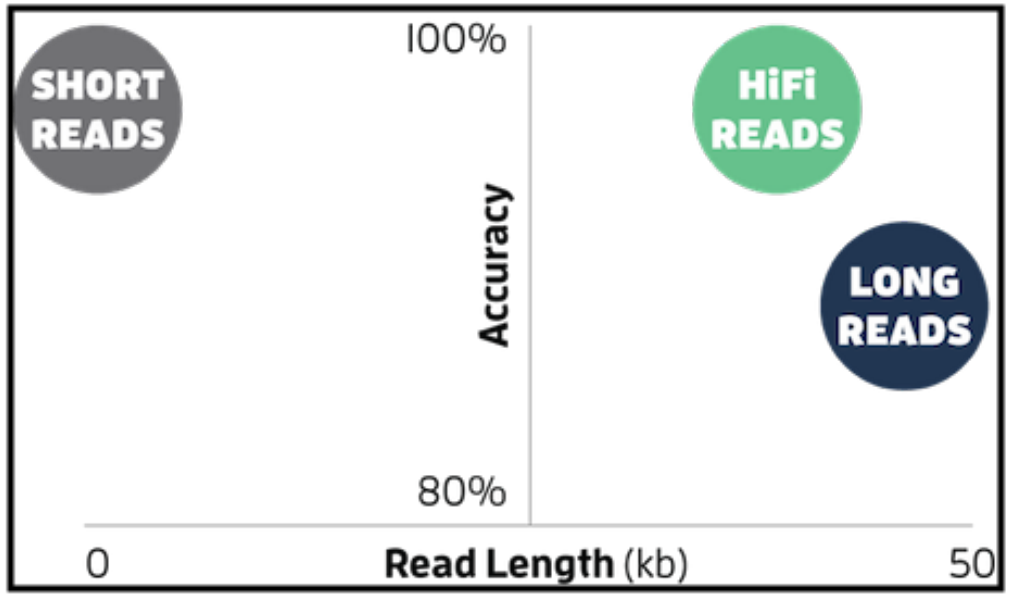
= reads

= physical DNA



A New Type of Read

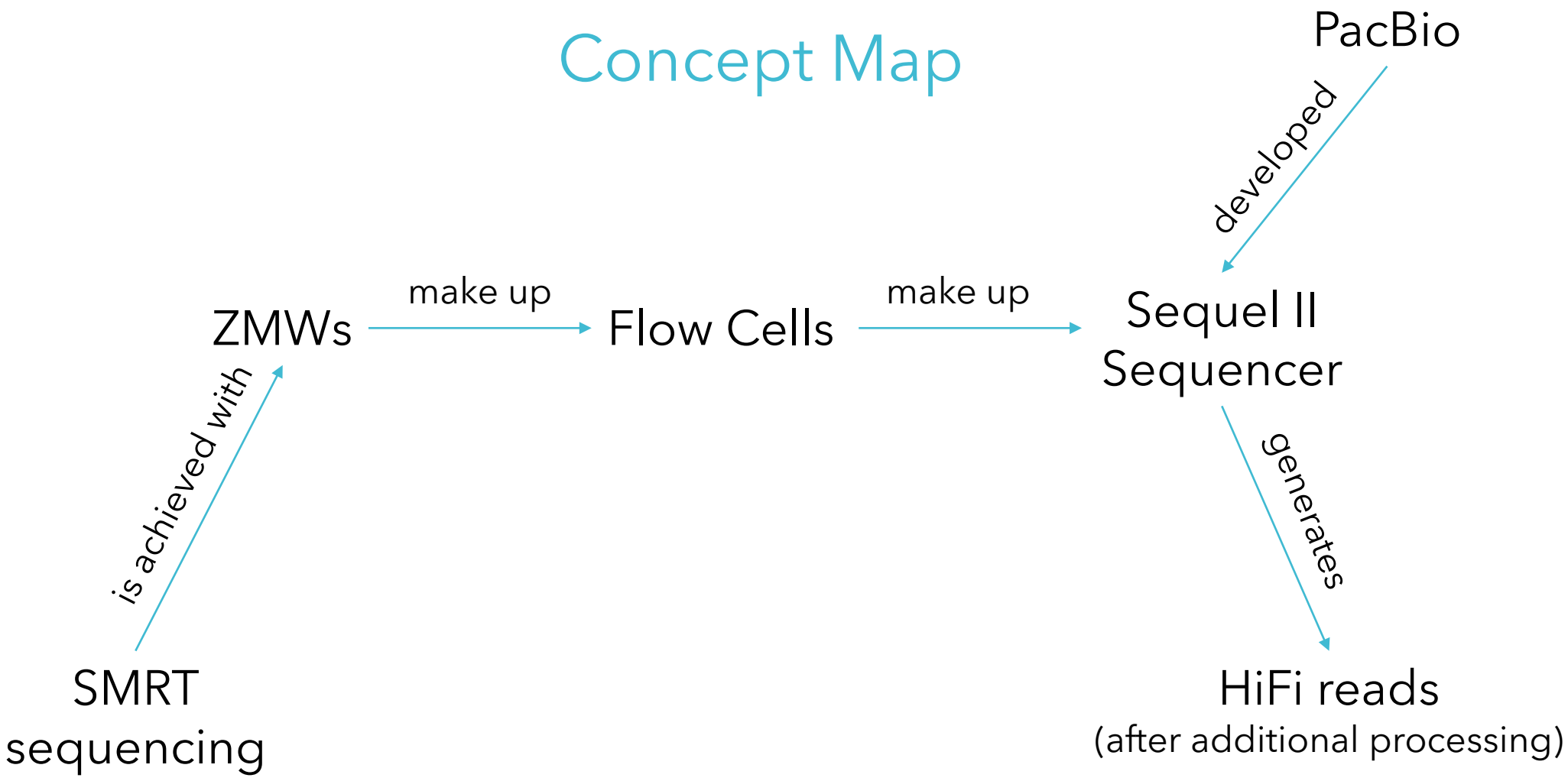
- Short reads = accurate but incomplete
- Long reads = complete but inaccurate
- Solution: HiFi



HiFi Technology & Vocabulary

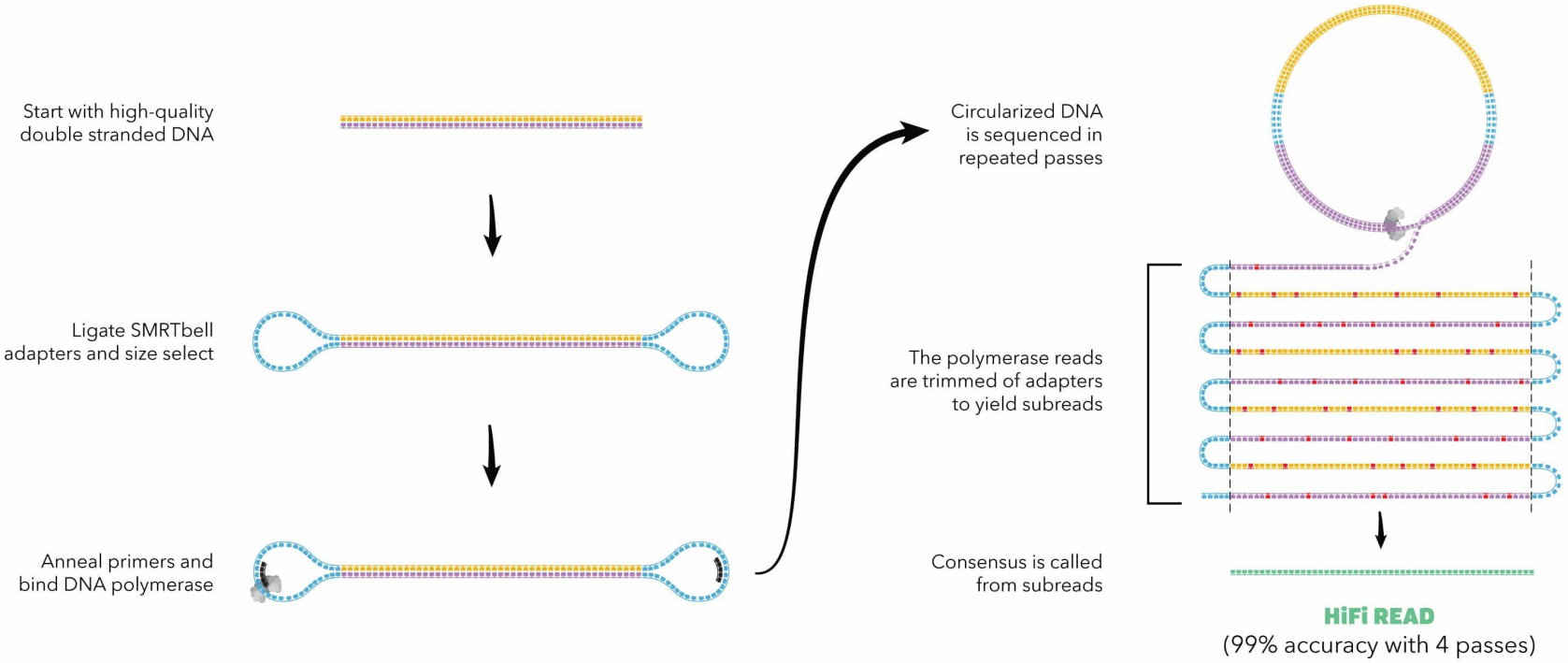
- HiFi = high fidelity
- PacBio = Pacific Biosciences (the developers)
- SMRT technology = single molecule, real time
- Sequel II = system of flow cells for sequencing
- ZMW = zero-mode waveguide (wells within each flow cell)

Concept Map


PacBio
developed
ZMWs
make up
Flow Cells
make up
Sequel II Sequencer
is achieved with
generates
SMRT sequencing
HiFi reads
(after additional processing)

Circular Consensus Sequencing

Alternative to CLR:



Assemblers Considered

- Peregrine
- Falcon Unzip
- Wegnan
- Flye
- Redbean/Wtdbg
- NextDeNovo
- Shasta
- Hifiasm
- Canu
- IPA

Hifiasm = new, fast, fully designed for HiFi

IPA = new and developed by PacBio

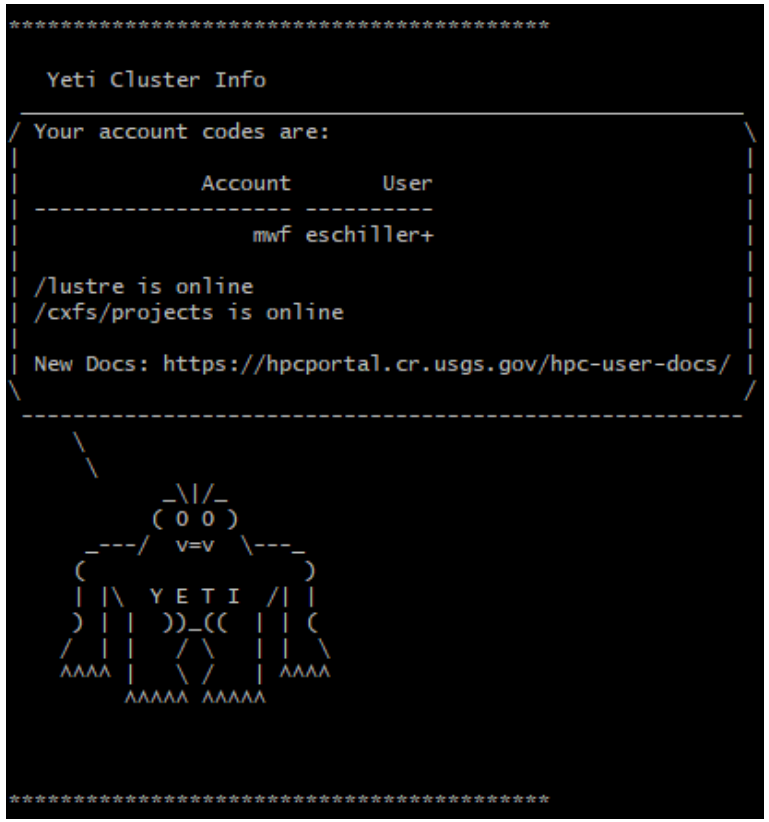
Canu = widely used

~ The Chosen Ones ~

The System

USGS Supercomputer = Yeti

- Based in Colorado
- Accessed via GitBash (or other terminal/simulator)
- Uses SLURM workload manager (Simple Linux Utility for Resource Management)
- Has lustre and cxfs file systems
- Uses 4 partitions (queues) for job submissions: normal, long large, UV



What does command-line programming look like?

- Command-Line Programming = submitting commands
  - often creating, manipulating, and moving around files/folders or running programs
- Basic bash commands:
  - mkdir = make directory
  - cd = change directory
  - mv = move or rename
  - nano = edit file
  - sbatch = submit script for running
  - rm = remove file
- Example visual:

```
(base) [eschillerstrom@yeti-login1 TESTcanu] rm -r !(test.*)
rm: remove write-protected regular file 'asm.seqStore/blobs.0001'? y
rm: remove write-protected regular file 'asm.seqStore/blobs.0002'? y
(base) [eschillerstrom@yeti-login1 TESTcanu] ls
test.sh  test.slurm
(base) [eschillerstrom@yeti-login1 TESTcanu] nano test.slurm
(base) [eschillerstrom@yeti-login1 TESTcanu] sbatch test.slurm
Submitted batch job 5747263
(base) [eschillerstrom@yeti-login1 TESTcanu] cd ~
(base) [eschillerstrom@yeti-login1 ~] ls
```

Change in Objective

Original Objective

Obtain genome
+
Upload to a database

→

Delay!

Black carp reads not available in time

→

New Objective

Automate pipeline and focus on usability

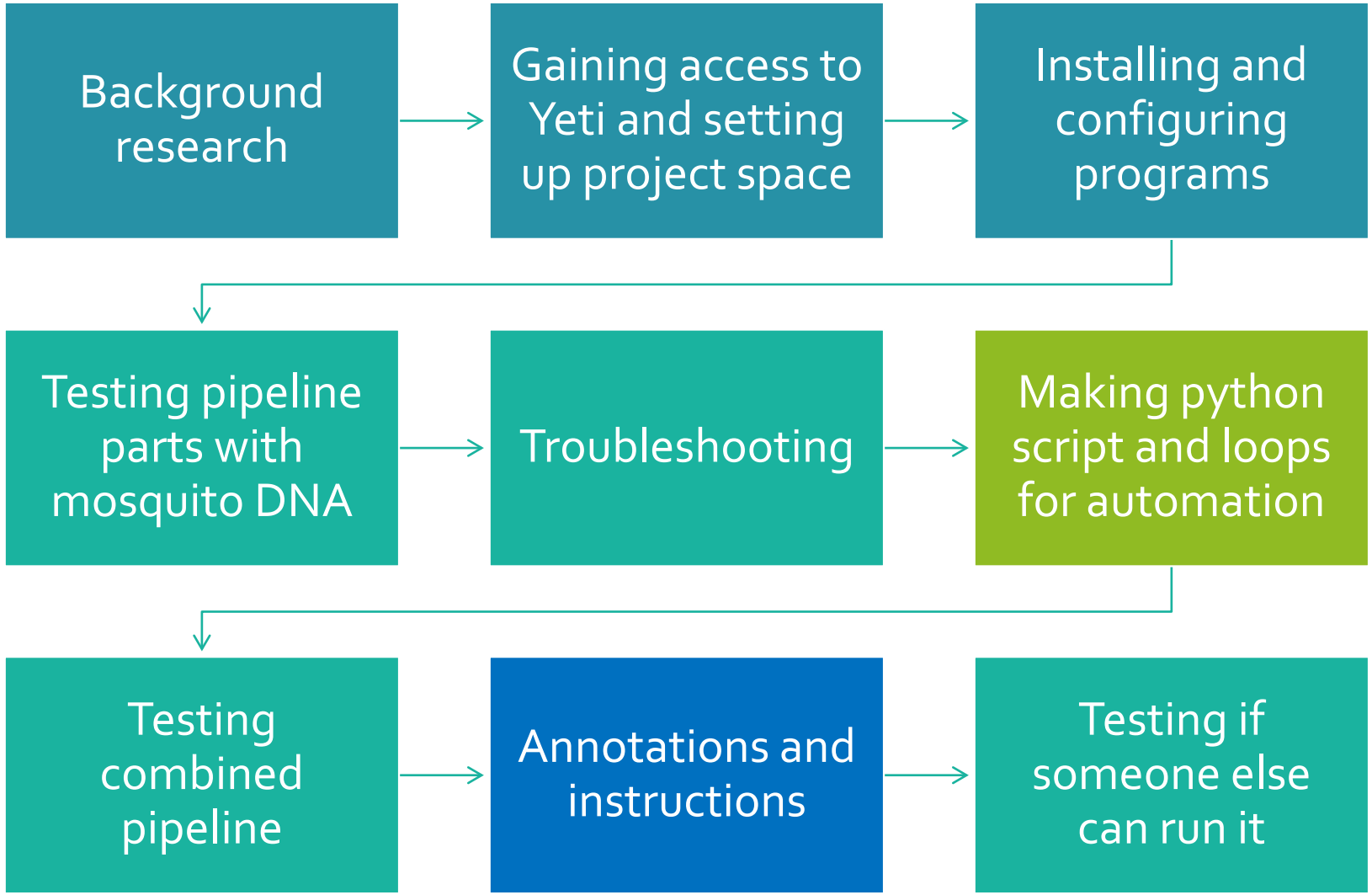
Timeline

Stage 1 – Research and set-up

Stage 2/4/6 – Testing and troubleshooting

Stage 3 – Automation and formatting results

Stage 5 – Finalizing and annotating



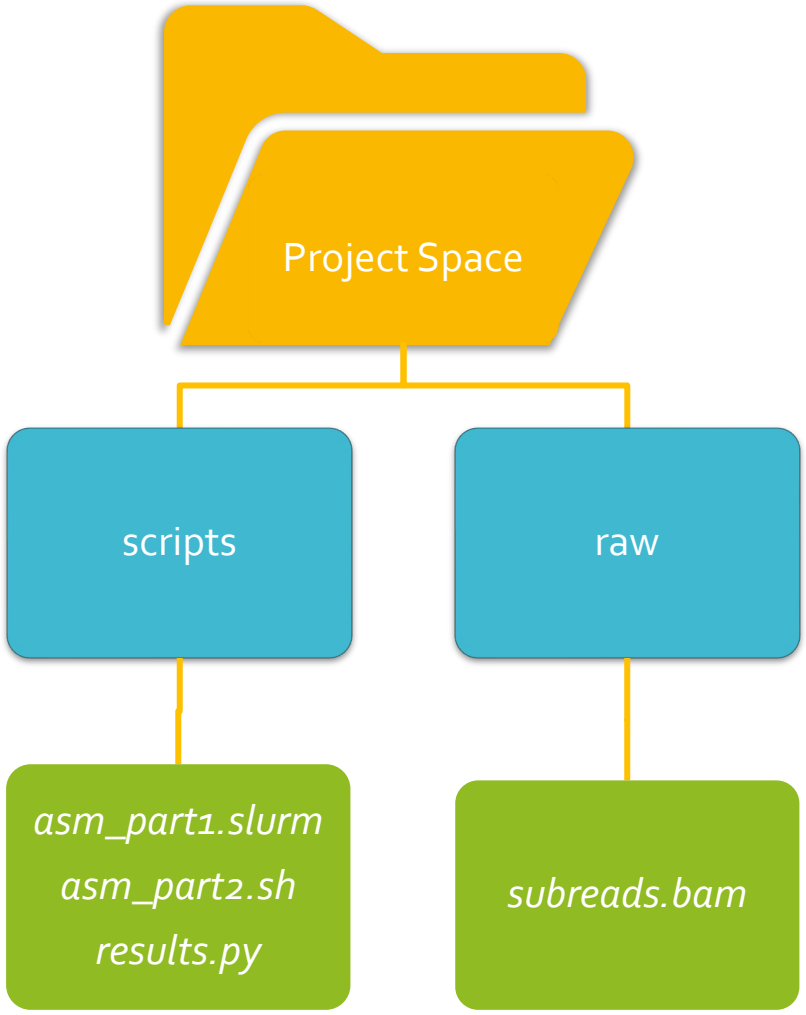
My Design

Inputs:

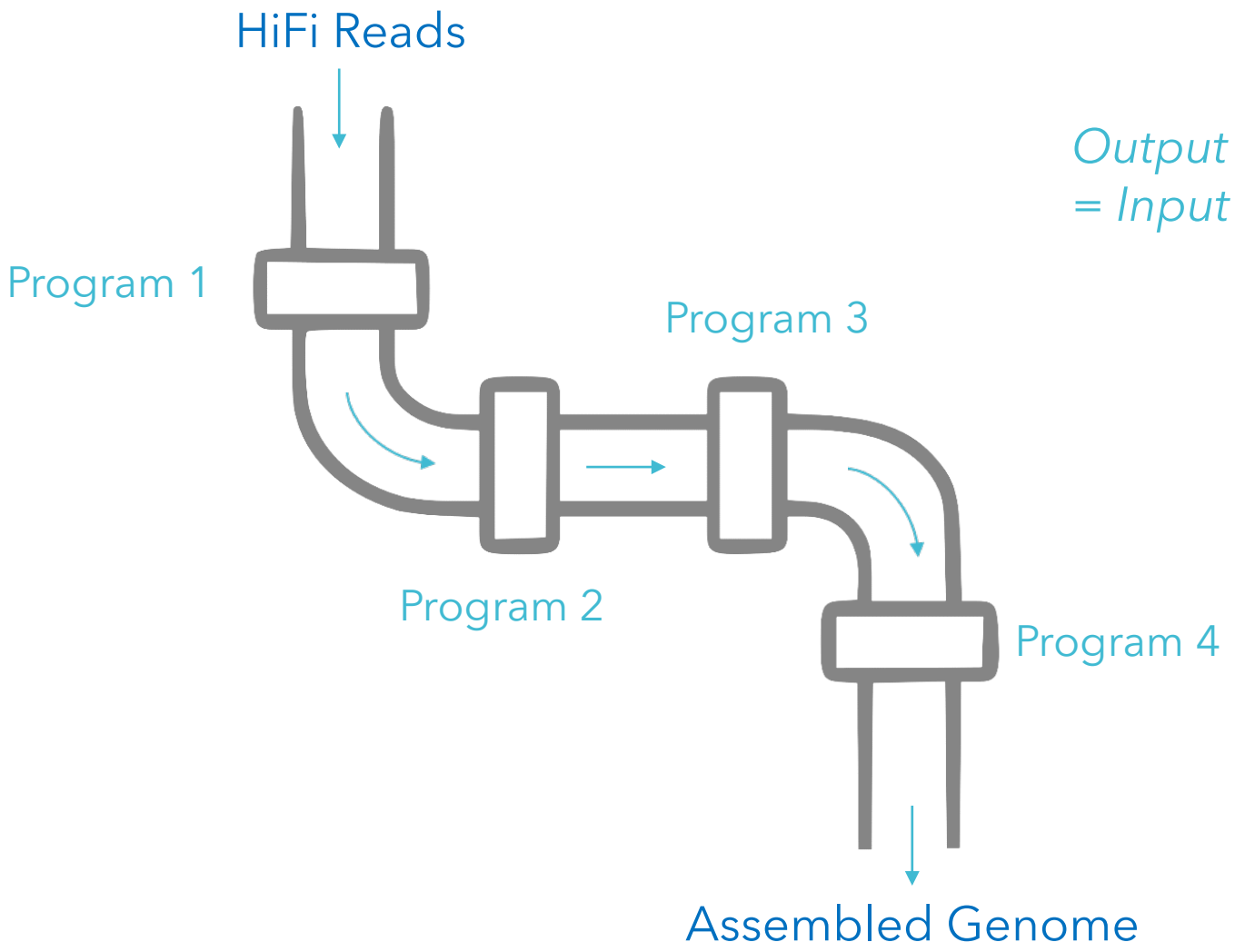
- job script of pipeline (.slurm)
- second half of script (.sh)
- script to generate output table (.py)
- raw subreads (.bam)

Outputs:

- quality report on reads (.html)
- 3 unique assemblies (.fasta)
- quality report on assemblies (.txt)


Project Space
scripts
raw
asm_part1.slurm
asm_part2.sh
results.py
subreads.bam

Pipeline Concept



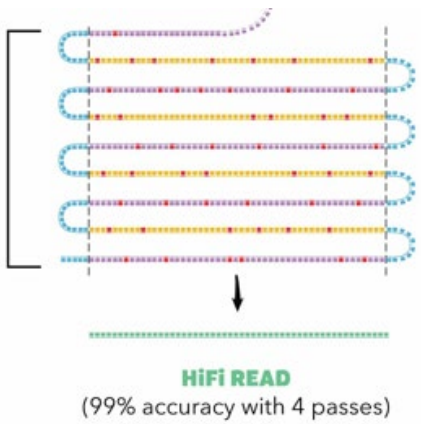
Output of each program
= Input for next

Pre-assembly

Pbccs:

Input: subreads → Output: HiFi reads

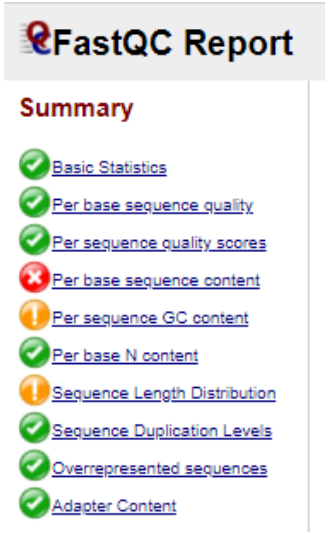
- Performs CCS processing
- Looks at subreads from the repeated passes → identifies errors
- Generates consensus



FastQC:

Input: HiFi reads → Output: Quality report

- Makes HTML file with graphs and metrics
- Ex. Phred scores, GC content, sequence lengths, etc.

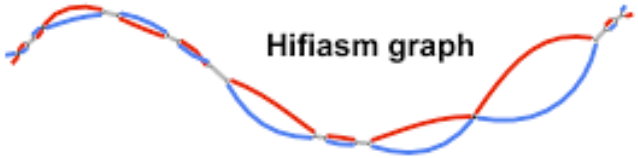


Assembly

Hifiasm:

Input: HiFi reads → Output: String graph

1. Overlap alignment
2. Error correction x3
3. Overlap alignment – round 2
4. String graph construction
5. Removal of redundant haplotigs



Note: extract fasta reads from gfa graph file

Assembly

IPA:

Input:
HiFi reads

→

Output:
Fasta file

1. Building Sequence Databases
2. Fast Overlap
3. Phase Separation – skipped!
4. Chimera and Repeat Filtering
5. Layout
6. Polishing

IPA
IMPROVED
PHASED
ASSEMBLY

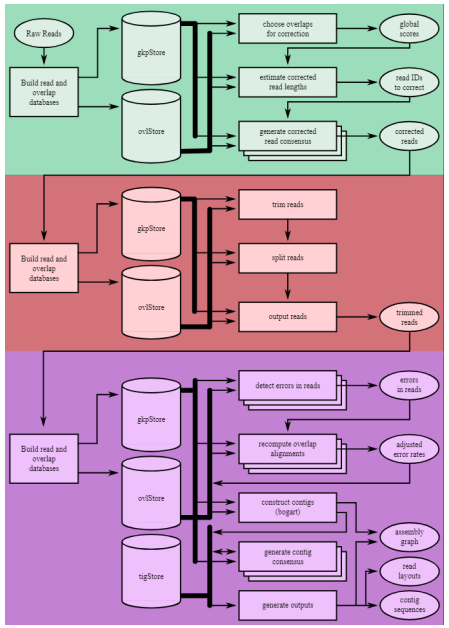
Assembly

Canu:

Input: HiFi reads → **Output:** Contig file

1. Database construction (repeated for each step)
2. Correction
3. Trimming
4. Assembly

Note: must include genome size and grid options in parameters



Purge_dups:

Input: Contig file & fofn → **Output:** Contig file

- Removes redundant haplotigs + heterozygous overlaps
- fofn = file of file names w/ path to raw reads

Post-assembly

QUAST:

Input: Assembly → Output: Text file

- Provides basic statistics on the assembly

```
All statistics are based on contigs of size >= 500 bp, unless otherwise noted
(e.g., "# contigs (>= 0 bp)" and "Total length (>= 0 bp)" include all contigs)
.

Assembly                    hifiasm
# contigs (>= 0 bp)         319
# contigs (>= 1000 bp)      319
# contigs (>= 5000 bp)      319
# contigs (>= 10000 bp)     318
# contigs (>= 25000 bp)     293
# contigs (>= 50000 bp)     185
Total length (>= 0 bp)      276124915
Total length (>= 1000 bp)   276124915
Total length (>= 5000 bp)   276124915
Total length (>= 10000 bp)  276115478
Total length (>= 25000 bp)  275577125
Total length (>= 50000 bp)  271662566
# contigs                   319
Largest contig              24831483
Total length                276124915
GC (%)                      44.04
N50                         9045660
N90                         783185
L50                         9
L90                         52
# N's per 100 kbp           0.00
```

Post-assembly

Meryl:

Input: HiFi reads → Output: Meryl file

- Creates file of k-mer counts from reads

Note: specify optimal k-mer size

Merqury:

Input: Assembly & meryl file → Output: Several files with metrics

- Uses k-mer counts to evaluate assembly completeness
- Compares counts in the initial reads to the counts in the final assembly

```
hifiasm all      208981442      229601001      91.0194
hifiasm 6799     276119173      58.874   1.29598e-06
```

Post-assembly

Results.py:

Input:
Assessment output files

→

Output:
Text file with table

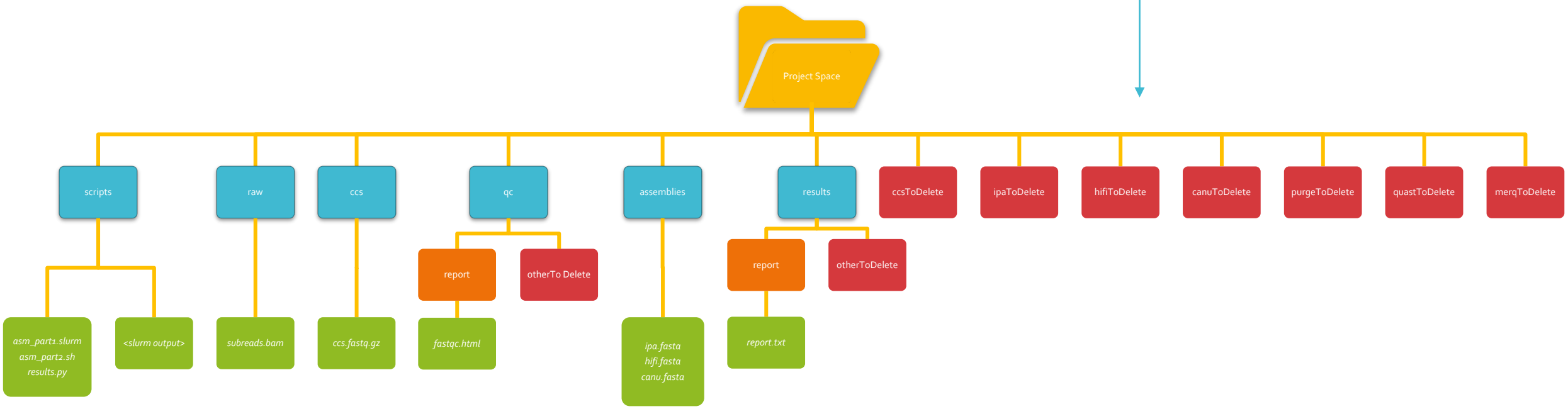
- Parses through 1 QUAST and 2 Merqury files per assembler
  - Total of 9 files
- Creates metrics table
- Compares assembly performance
- Not installed program

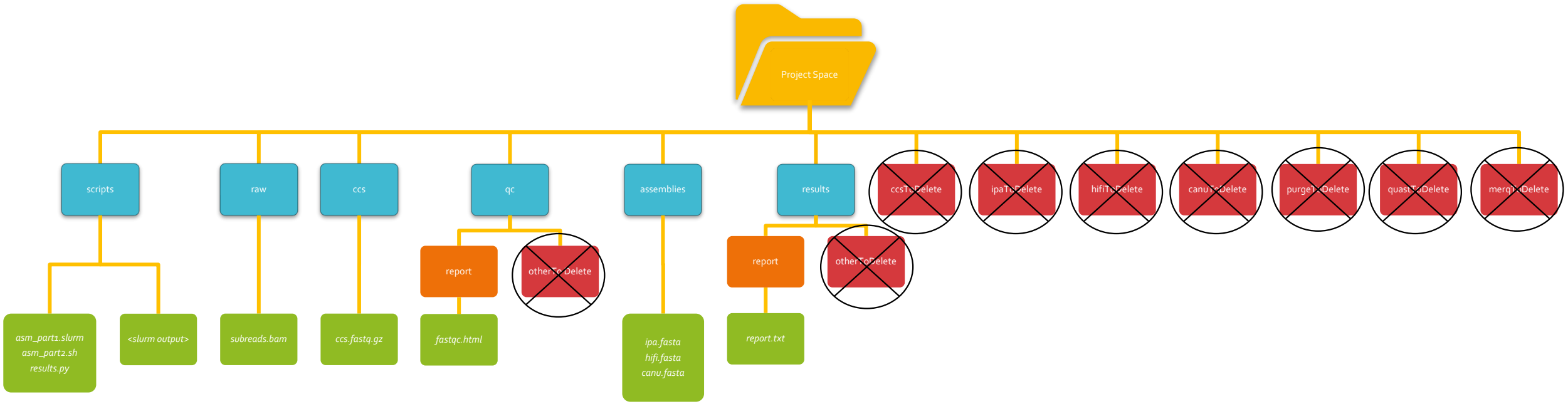
(visual of table later)

The Pipeline

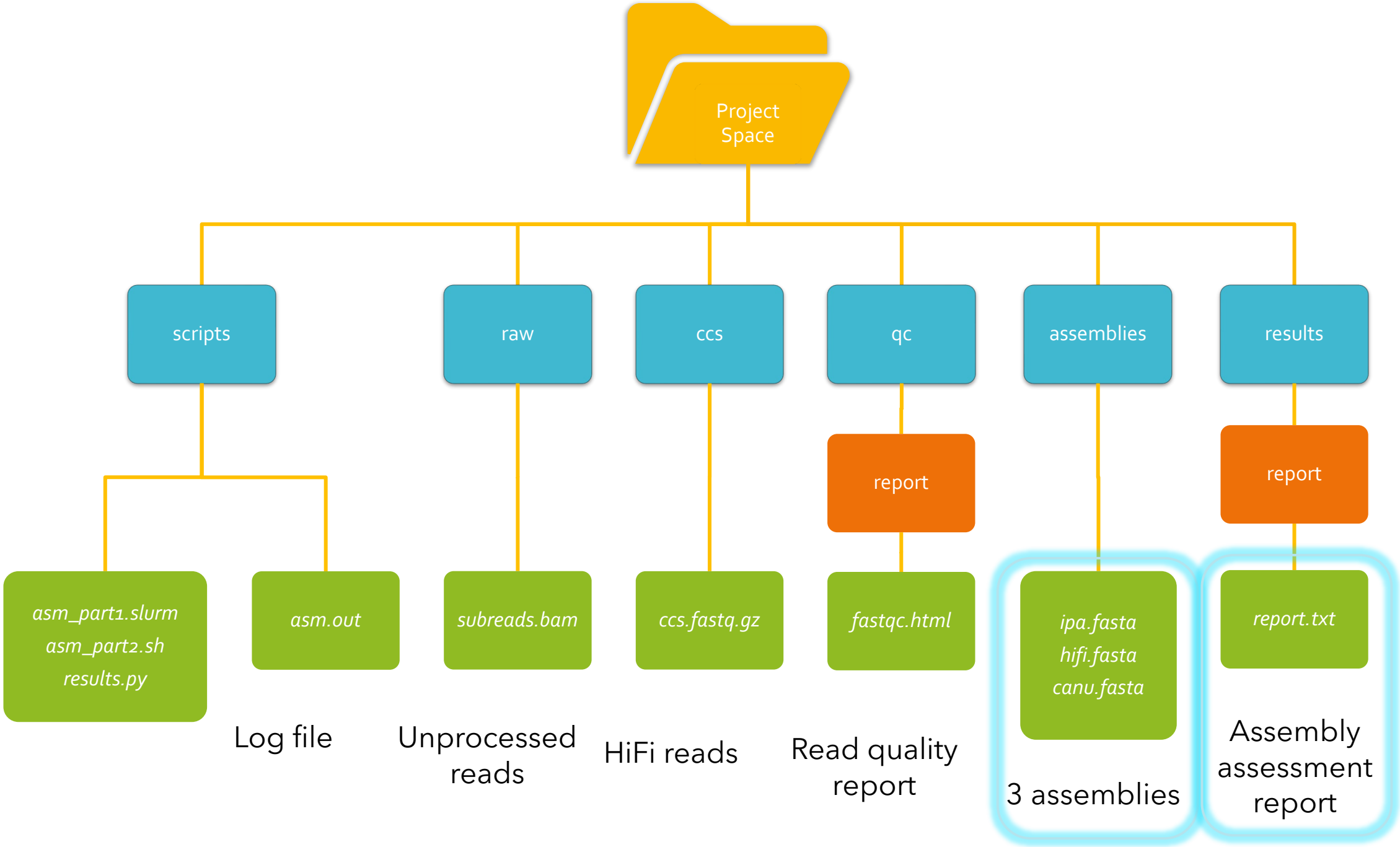
- Pre-assembly:
  - Pbccs → HiFi from raw reads
  - FastQC → Quality report
- Assembly:
  - Hifiasm → Assembly 1
  - IPA → Assembly 2
  - HiCanu & purge_dups → Assembly 3
- Post-assembly:
  - QUAST → Size and contiguity metrics
  - Meryl & Merqury → Completeness and quality metrics
  - Results.py (self-generated) → Table of all wanted metrics

Format: <program>ToDelete






```
rm -r *ToDelete
```


Project Space
scripts
raw
ccs
qc
assemblies
results
report
report
asm_part1.slurm
asm_part2.sh
results.py
asm.out
subreads.bam
ccs.fastq.gz
fastqc.html
ipa.fasta
hifi.fasta
canu.fasta
report.txt
Log file
Unprocessed reads
HiFi reads
Read quality report
3 assemblies
Assembly assessment report

Assessment Metrics

The Four C's:

- Contiguity ↑ - want little fragmentation
- Correctness ↑ - want high base accuracy
- Completeness ↑ - want few missing bases/fragments
- Computation ↓ - want small file sizes and fast analysis time

Output table

Contiguity

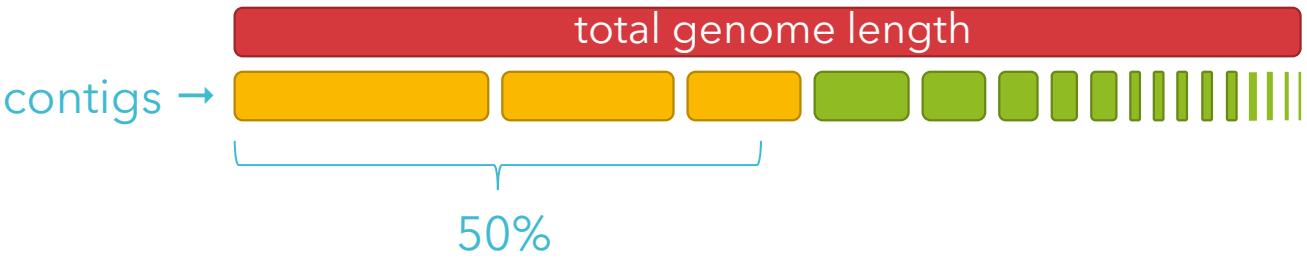
Metrics	Hifiasm	HiCanu	IPA	Recommendation
Assembly Size (bp)	276,124,915	238,404,105	237,588,416	n/a
Number of Contigs	319	337	256	ipa
Longest Contig (bp)	24,831,483	7,202,325	7,248,392	hifiasm
N50	9,045,660	2,092,475	2,086,592	hifiasm
L50	9	32	35	hifiasm
K-mer Completeness (%)	91.0194	86.8194	89.072	hifiasm
Quality (QV)	58.874	63.2767	51.0179	canu

Correctness

Completeness

Metric Breakdown

- Contig = fragment of contiguous sequence



- N50 = shortest contig length among fewest to make up ½ of genome
- L50 = fewest number of contigs to make up ½ of genome
- Completeness = % k-mers in input also found in assembly
- QV = marks k-mers found only in assembly as errors & uses this error rate in Phred score formula

Snapshots of Annotated Script

Black Carp De Novo Genome Assembly Pipeline with Multi-Assembler Comparison

By Emma Schillerstrom

Directions:

- ***Key for this document:***
  - ***Green highlight = pre-assembly step***
  - ***Yellow highlight = assembly step***
  - ***Blue highlight = post-assembly step***
  - ***Grey highlight = what individual user MAY NEED TO EDIT***
  - ***Red font = commented lines in script***
  - ***Purple font = NOT part of a script***
  - ***Red highlight = varies by species***
- Required installations:
  - Miniconda3, Canu, Improved Phased Assembly (IPA), Hifiasm, purge_dups, pbccs (Bioconda package), FastQC, QUAST, Meryl, Merqury
  - Dependencies: runner, minimap2

- Setting up project space:
  - Obtain project space by emailing gs-css_csas_hpc_help@usgs.gov and requesting space on the lustre system (unless space already exists)
  - Optional – make variable to go directly to your project space
    - `cd ~ && nano .bash_profile`
    - Add alias below exported path variable
    - Example: `alias carp="cd /lustre/projects/fws/fws_mwf/EMS"`
    - `source .bash_profile`
  - Make folders for the scripts and raw starting data
    - Scripts directory:
      - `carp` (see alias above)
      - `mkdir scripts && cd scripts`
      - `nano asm_part1.slurm`
        - Copy and paste script from text file
        - Update sections that are highlighted in grey or red in this document
        - Save and quit
      - `nano asm_part2.sh`
        - Copy and paste script from text file
        - Update sections that are highlighted in grey or red in this document
        - Save and quit
        - Run: `chmod +x asm_part2.sh`
        - When you "`ls`" in the scripts folder, this file should now be green, meaning it is executable

```
#SBATCH --job-name=TotalAssembly
#SBATCH -p normal
#SBATCH -n 5
#SBATCH -N 4
#SBATCH --account=mwf
#SBATCH --time=7-00:00:00
#SBATCH --mail-type=ALL
#SBATCH --mail-user=emma_schillerstrom@fws.gov
#SBATCH -o job_assembly-%j.out

module load java/jdk-1.8.0_162
module load zlib/1.2.11

#PRE-ASSEMBLY

#CCS

#Makes execution directory
cd .. && mkdir ccsToDelete && cd ccsToDelete
#Generates consensus reads from PacBio subreads
ccs --maxLength <based on library insert size> --minPasses 3 --num-threads 240
/lustre/projects/fws/fws_mwf/raw/*.subreads.bam ccs.fastq.gz
#Puts them in a new folder for easy access
mkdir ../ccs
cp *.fastq.gz ../ccs

#FastQC

#Makes execution directory
cd .. && mkdir qc
cp ccs/*.fastq.gz qc && cd qc
#Generates quality control report on reads
fastqc *.fastq.gz
#Organizes files into html "report" and "other"
shopt -s extglob
mkdir otherToDelete && mv !(otherToDelete) otherToDelete
mkdir report && mv otherToDelete/*.html report

#ASSEMBLY

#Hifiasm

#Makes empty folder for assemblies
cd .. && mkdir assemblies
#Makes execution directory, grabs reads, and changes extension
mkdir hifiToDelete && cd hifiToDelete
cp ../ccs/*.fastq.gz .
mv *.fastq.gz carp.fq.gz
```

My Deliverables

- Bash script part 1 (.txt)
- Bash script part 2 (.txt)
- Results python script (.txt)
- User guide doc (.docx)

Steps for Use on Different Yeti account

1. Request account & project space
2. Perform installations:
   - pbccs, FastQC, hifiasm, canu, purge_dups, IPA, QUAST, meryl, merqury, runner
3. Set up folder with proper scripts
4. Replace highlighted sections
5. Run

Strengths of My Design

- ✓ Optimized from experience running other pipelines
- ✓ Well-organized outputs
  - ✓ Easy to find desired files
  - ✓ Difficult to delete critical files
  - ✓ Easy troubleshooting
- ✓ 3 assemblies = options
  - ✓ Comparisons and recommendations offered to save time
- ✓ More than a job script
  - ✓ Instructions
  - ✓ Colored coding
  - ✓ Annotations
  - ✓ Folder visuals

Acknowledgements

- Supervisors: Katherine Bockrath, Zebadiah Woiak
- Nathan Whelan
- Pacbio Help Team
- Yeti Help Team
- Whitney Genetics Lab & Midwest Fisheries Center
- Directorate Fellows Program Coordinators

Questions?