An Engaging CS1 Curriculum Using BRIDGES

Matthew Mcquaigue*, Allie Beckman*, David Burlinson*, Luke Sloop*, Alec Goncharow*, Erik Saule*, Kalpathi Subramanian*, and Jamie Payton+

{mmcquaig, abeckma2, dburlins, lsloop4, agoncha1, esaule, krs}@uncc.edu, payton@temple.edu

*UNC Charlotte. +Temple University

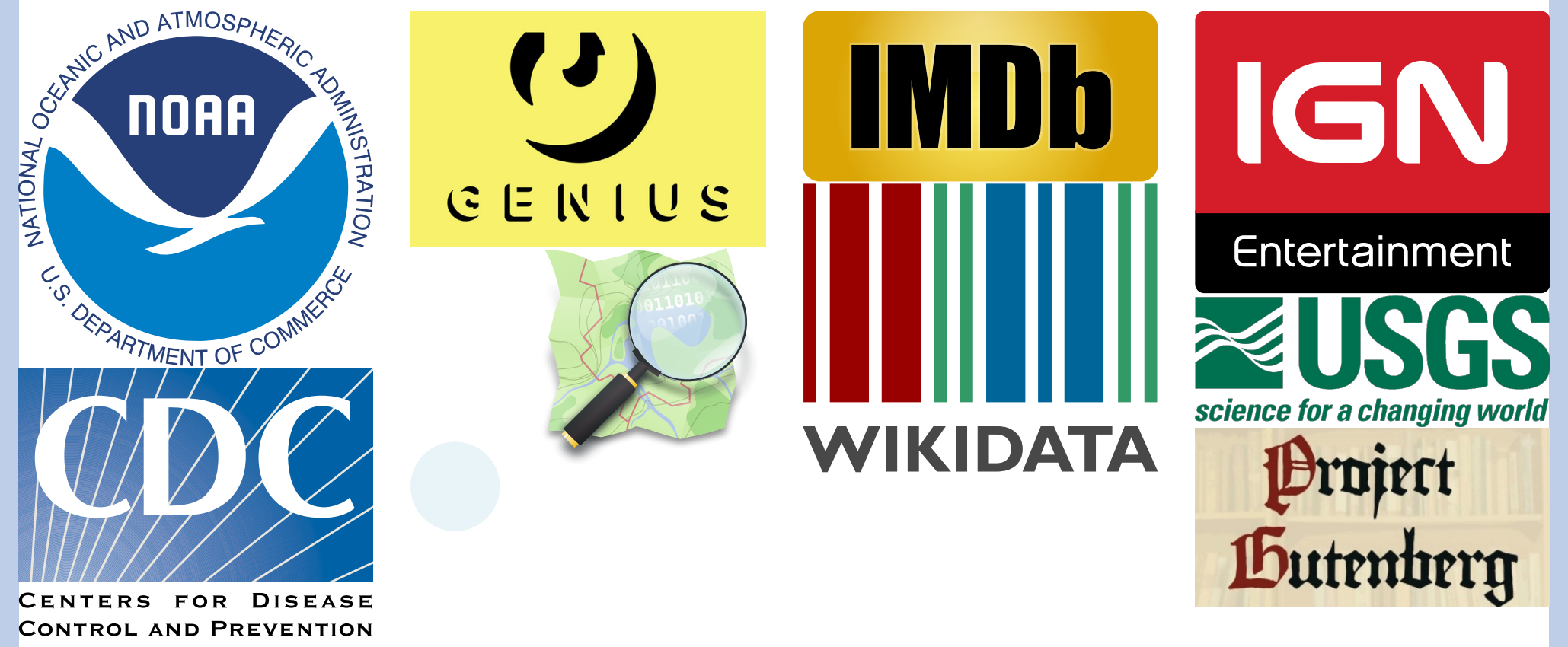
Motivation

With growing student populations, the typical CS1 student now sees the major as a way to a career rather than as a calling. As such, there is a need to engage CS students in programming while still imparting the necessary knowledge at that level. Many CS1 courses tend to focus on terminal based assignments that are not engaging enough anymore. Many others feel the pressure to dilute the technical content to engage and retain majors.

BRIDGES Mantra

Computer Science does not have to be boring to be rigorous: Good dataset and visualization can engage students without sacrificing rigor.

BRIDGES Data Sources


NOAA
NATIONAL OCEANIC AND ATMOSPHERIC ADMINISTRATION
U.S. DEPARTMENT OF COMMERCE
GENIUS
IMDb
WIKIDATA
IGN
Entertainment
USGS
science for a changing world
CDC
Centers for Disease Control and Prevention
Project Gutenberg

Using BRIDGES in your course

- BRIDGES can freely be used by anyone.
- BRIDGES is appropriate for use in CS1, CS2, Data Structure, and Algorithm courses.
- The students can program in either C++, Java, or Python.

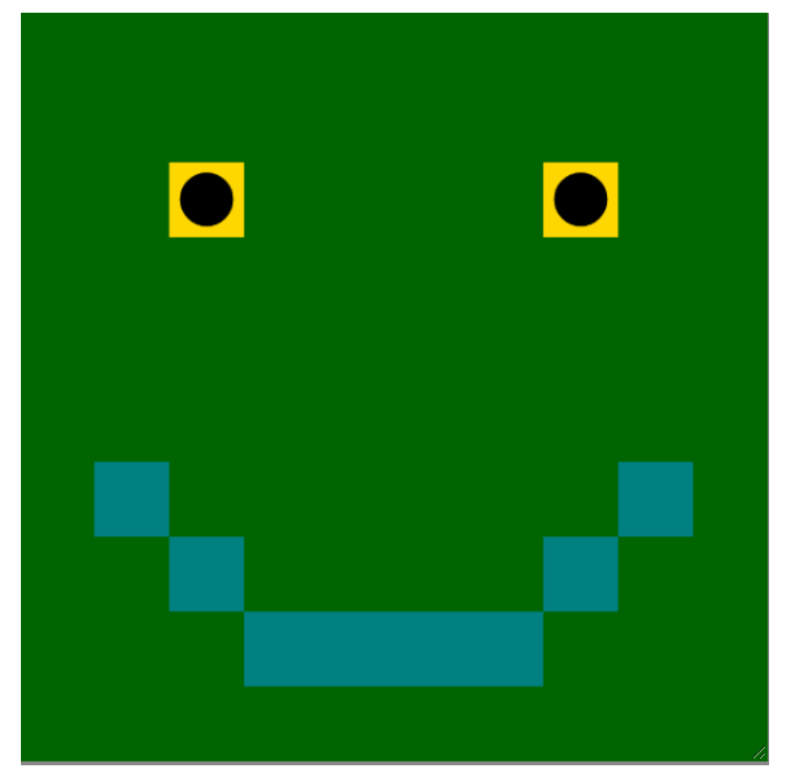
BRIDGES Game API (here in Java)

```java
import bridges.games.*;
import bridges.base.*;

class InputGame extends NonBlockingGame {
    public InputGame(int assign, String username, String apikey) {
        super (assign, username,
apikey, 10, 10);
    }

    public void initialize() {
        for (int i=0; i<getBoardHeight(); ++i)
            for (int j=0; j<getBoardWidth(); ++j)
                setBGColor(i, j, NamedColor.ivory);
    }

    public void gameLoop() {
        //This function is executed each frame of the game
        if (keyUp()) {
            drawSymbol(1, 1, NamedSymbol.U, NamedColor.blue);
            drawSymbol(1, 2, NamedSymbol.N, NamedColor.yellow);
            drawSymbol(1, 3, NamedSymbol.C, NamedColor.black);
            drawSymbol(1, 4, NamedSymbol.C, NamedColor.green);
        }
        else {
            drawSymbol(1, 1, NamedSymbol.none, NamedColor.blue);
            drawSymbol(1, 2, NamedSymbol.none, NamedColor.yellow);
            drawSymbol(1, 3, NamedSymbol.none, NamedColor.black);
            drawSymbol(1, 4, NamedSymbol.none, NamedColor.green);
        }
    }

    public static void
main (String args[]) {
    InputGame g = new InputGame(ASSIGNMENT_NUMBER, "YOUR_USER_ID",
                                "YOUR_API_KEY");
        g.start();
    }
}
```

BRIDGES Data Visualization API (here in C++)

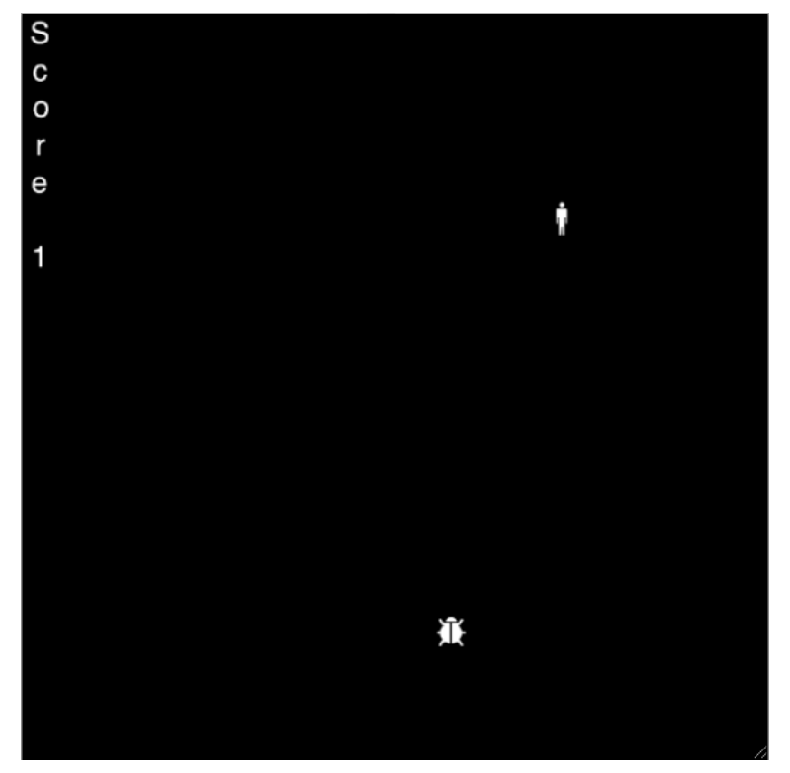
```cpp
#include "Bridges.h"
#include "ColorGrid.h"
#include "DataSource.h"
#include "data_src/ElevationData.h"
using namespace bridges;

int main(int argc, char **argv) {
  // initialize Bridges
  Bridges bridges(123, "BRIDGES_USER_ID", "BRIDGES_API_KEY");
  DataSource ds (bridges);
  ElevationData *elev_data = ds.getElevationData(6.02,
                              46.10, 8.70, 50.77, 0.02);

  // create color grid
  int width = elev_data->getCols();
  int height = elev_data->getRows();
  int maxVal = elev_data->getMaxVal();

  ColorGrid cg(height, width);
  // load the elevation data
  for (int j = 0; j < height; j++)
    for (int k = 0; k < width; k++) {
      float pixel_val = (float) elev_data->getVal(j, k);
      if (pixel_val < 0)
        cg.set(j, k, Color(255, 0, 255));
      else {
        // scale value to be within 0-255, for r,g,b range
        int gray = (int) (pixel_val * 255. / maxVal);
        cg.set(j, k, Color(gray, gray, gray));
      }
    }

  // visualize
  bridges.setDataStructure(&cg);
  bridges.visualize();

  return 0;
}
```

Etch-A-Smiley



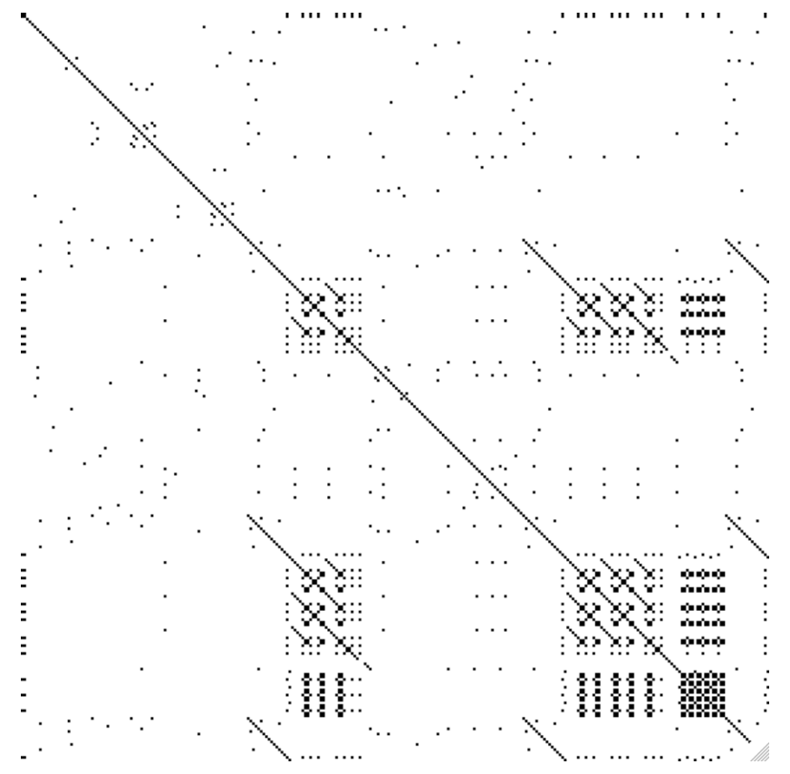
Description: Draw a Smiley using the BRIDGES Game API. You can style it whichever way you like.
Topics: API Call, 2D coordinates
Engagement: Visual, Creative

Bugstomp



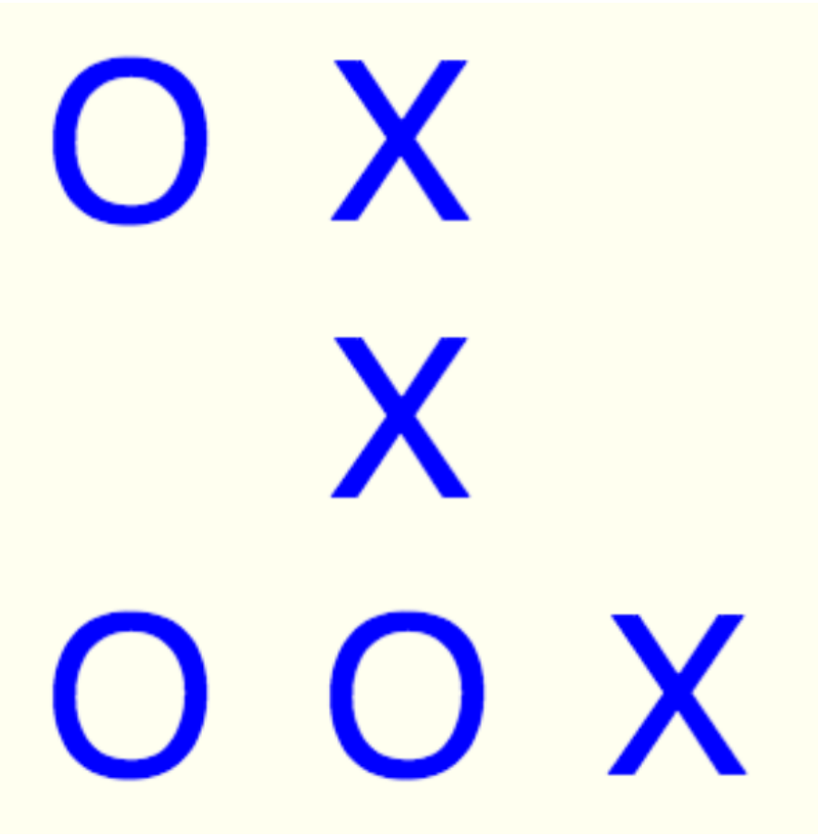
Description: Move the character to stump a randomly located bug. Keep track of score.
Topics: Conditionals, 2D coordinates, Input-Process-Output, Random Numbers
Engagement: Interactive Game

Song Lyrics



Description: Show the repetition in the lyrics of a song by coloring the (i, j) pixel black if words i and j are the same.
Topics: Nested for-loops, Conditionals, String Operations
Engagement: Visualization, Real Lyrics Data

Tic Tac Toe

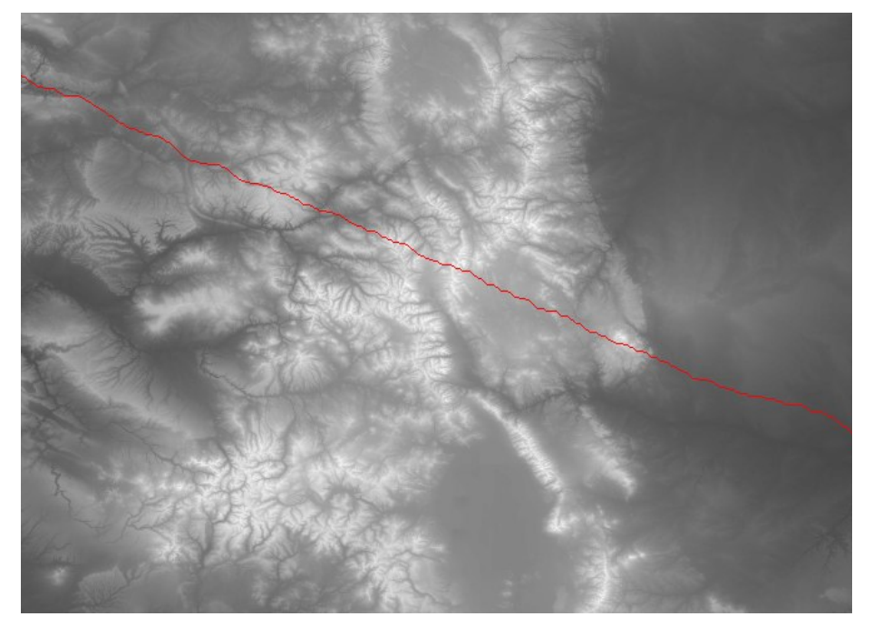


Description: A classic board game with 2-player mode, and 1-player mode with a mostly random AI.
Topics: Conditionals, Input-Process-Output, 2D coordinate, for Loops, Simple AI
Engagement: Interactive Game, Basic AI

Image Processing

Description: Implement basic image manipulation algorithms like grayscaling, component filtering, flipping, and blending.
Topics: File Input, Image Representation, Nested for-Loops
Engagement: Basic Image Filters

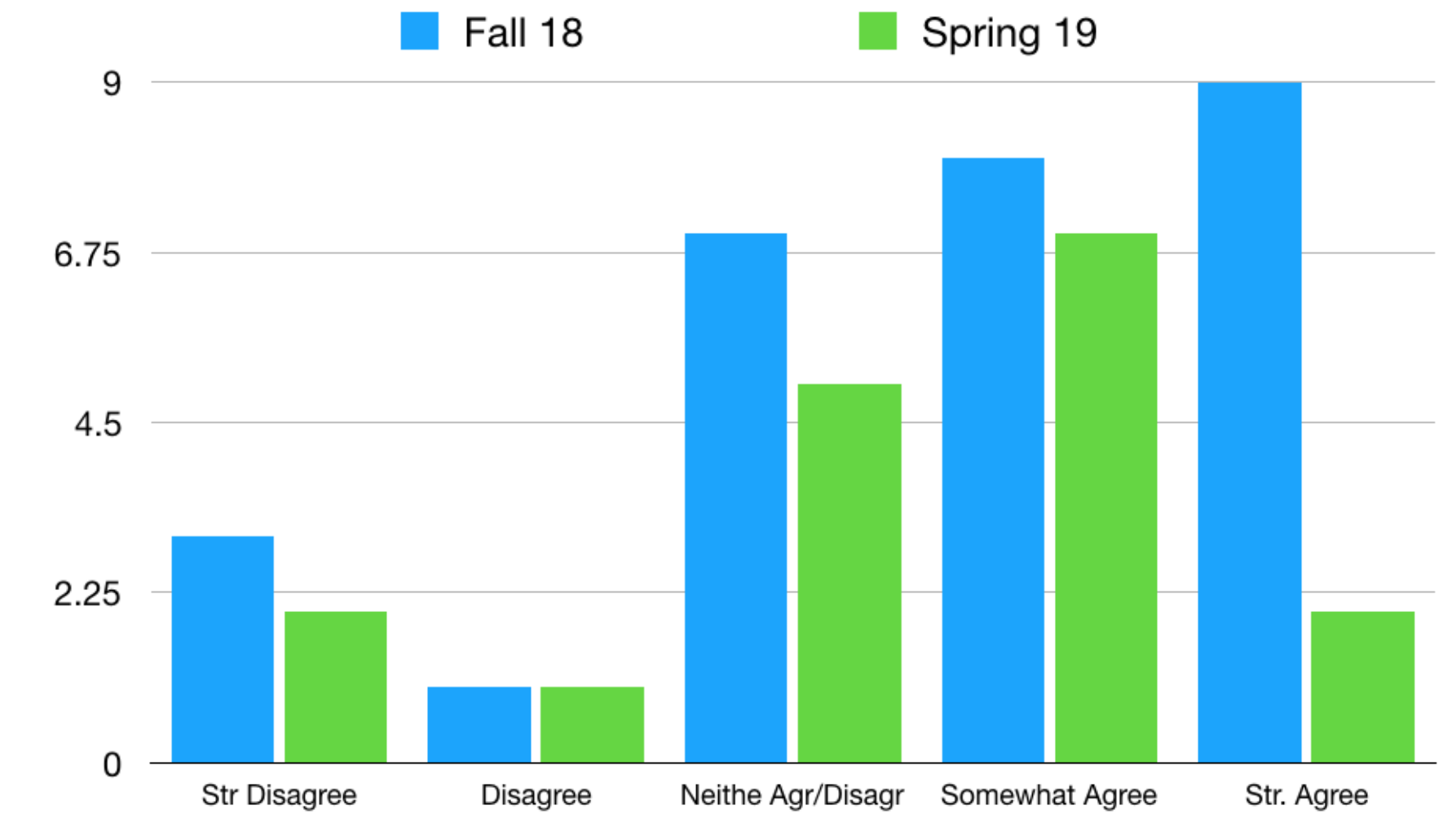
Mountain Path (Nifty Assignment)



Description: Find a path of least effort walking through a mountain.
Topics: for-loops, Conditional, Greedy Strategies, Edge Cases
Engagement: Real-world Data, Visualization

Student feedback from some early CS courses

- "really liked seeing how easily implement some simple image processing with this assignment"
- "excellent practical example of greedy algorithm"
- "feel challenged, but also feel satisfied"
- "liked the assignment, enjoyed creating the game"
- "allows creativity in a new way!"



Mountain path increased my interest in computing.

Opportunities for instructors

- The BRIDGES team is sponsoring instructors to use BRIDGES in their courses to gather data on student feedback.
- We are looking for instructors of CS1, CS2, Data Structure, and Algorithms courses.
- The BRIDGES team will provide technical support and assignment support and will help with designing new assignments.
- Contact us if interested!

Acknowledgment

This work was partially supported by the National Science Foundation grant DUE-1726809.