Pixel

Alex Anthony Cortes-Ose
Dillon Davis
Jessica Kim
Jessica Peng
AGENDA

01
Team Introduction

02
Our Inspiration

03
Pixel Language Details

04
Syntax and Grammar, Types and Operators, Built-in Functions

05
Open CV

06
Test Suites

07
Demo
Team

Alex Anthony Cortes-Ose  
Language Guru

Dillon Davis  
Manager

Jessica Kim  
Tester

Jessica Peng  
System Architect
Our Inspiration

- Computer vision: automated image-editing
- Image editing
  - Instagram
  - Photo-editing apps
- Pixel manipulation
Pixel Language Details

- Designed to process and manipulate images
- Can create filters and perform various tasks fundamental to basic image processing
- Images are represented by matrices
- Similar {bracket} syntax and type declaration as Java
- Similar matrix manipulation and lists as Python
- Types: int, str, float, matrix, void
- Keywords: if, else, while, for, fun, return, print
```cpp
fun::void main() {
    image base = image_in("../img/flowers.JPG");
    matrix::float base_gray = base.grayscale;
    matrix::float final = zeros(base_gray.rows, base_gray.cols);
    int i;
    for (i = 0; i < base_gray.rows; i = i + 1) {
        int j;
        for (j = 0; j < base_gray.cols; j = j + 1) {
            if (base_gray[i, j] >= 0.58) {
                final[i, j] = 1;
            }
        }
    }
    image_out("../img/flower_centers.JPG", final);
}
```

### Syntax & Grammar

- **function declaration + return types**
- **typed variables**
- **matrix cols and rows**
- **matrix indexing**
- **Python-style matrix literals**

```cpp
sobel_horizontal = [
    [1, 0, -1],
    [2, 0, -2],
    [1, 0, -1]];
```
### Types and Operators

**Matrix multiplication:**

```cpp
matrix::float A = [
    [-1, 4],
    [2, 3]
];
matrix::float B = [
    [9, -3],
    [6, 1]
];
matrix::float C = A * B;
```

**Image types** have red, green, blue, and grayscale 2d Float Matrices.

**Matrix types** have rows, columns and a matrixAddr 2D float matrices.
**Built-in Functions**

- `image_in` -> takes in [String; String] and return Image
- `Image_out` -> takes in [String; Image; String] and returns Void
- `convolute` -> takes in [Matrix; Matrix] and returns Matrix
- `join` -> takes in [Matrix; Matrix; Matrix] or [Matrix] and returns Image
struct matrix {
    int rows;
    int cols;
    float** matrixAddr;
};
struct image {
    matrix* red;
    matrix* green;
    matrix* blue;
    matrix* grayscale;
};

matrix* initMatrix(float* data, int num_rows, int num_cols) {
    Mat newMatrix = Mat::zeros(num_rows, num_cols, CV_32F);
    for (int i = 0; i < (num_rows * num_cols); i++) {
        newMatrix.at<float>(i) = data[i];
    }
    matrix* result = (matrix*) malloc(sizeof(struct matrix));
    result->cols = num_cols;
    result->rows = num_rows;
    result->matrixAddr = getMat(newMatrix);
    return result;
}

Mat GS = Mat::zeros(grayscale->rows, grayscale->cols, CV_32F);
for (int i = 0; i < grayscale->rows; i++) {
    for (int j = 0; j < grayscale->cols; j++) {
        GS.at<float>(i, j) = grayscale->matrixAddr[i][j];
    }
}

image* result = (image*) malloc(sizeof(struct image));
result->grayscale = initMatrix(get1D(getMat(GS), GS.rows, GS.cols), GS.rows, GS.cols);
return result;
Code Samples
fun::void main() {
    image base = image_in("../img/cat.png");
    matrix::float box =
        [[1, 1, 1],
         [1, 1, 1],
         [1, 1, 1]];
    matrix::float fr = convolute(base.red, box) * 0.11;
    matrix::float fg = convolute(base.green, box) * 0.11;
    matrix::float fb = convolute(base.blue, box) * 0.11;

    image final = join(fr, fg, fb);
    image_out("../img/blurred-cat.png", final);
}
Before
After
Matrix Addition

fun void main() {
    matrix float A = [
        [1.0, 1.0],
        [1.0, 1.0]
    ];
    matrix float B = [
        [2.0, 3.0],
        [4.0, 5.0]
    ];
    print(A + B);
}
```java
fun::void main() {
    image base = image_in("../img/steam-engine.png");
    matrix::float base_gray = base.grayscale;
    matrix::float sobel_horizontal = 
        [[1, 0, -1],
         [2, 0, -2],
         [1, 0, -1]];
    matrix::float horiz = convolute(base_gray, sobel_horizontal);

    matrix::float sobel_vertical = 
        [[1, 2, 1],
         [0, 0, 0],
         [-1, -2, -1]];
    matrix::float vert = convolute(base_gray, sobel_vertical);

    matrix::float fhoriz = horiz * horiz;
    matrix::float fvert = vert * vert;
    matrix::float final = (fhoriz + fvert) ** (1/2);

    image_out("../img/steam-engine-edges.png", final);
}
```
Testing

- Test suites for syntax, grammar, functionality, etc.
- Tests for cases that should produce errors as well as sample program output
- Over 60 tests in the repository
Sample Test Suites

```java
fun::int main() {
    int i;
    for (i = 0; i < 10; i = j + 1) {} /* Error: j is undefined */
    return 0;
}
```

- Tests that an undefined variable raises an error
Sample Test Suites

```java
fun::int add(int a, int b)
{
    int c;
    c = a + b;
    return c;
}

fun::int main()
{
    int d;
    d = add(52, 10);
    print(d); /* Should print 62 */
    return 0;
}
```

- Tests that function calling, integer addition and printing work
Pixel Demo
Thank You