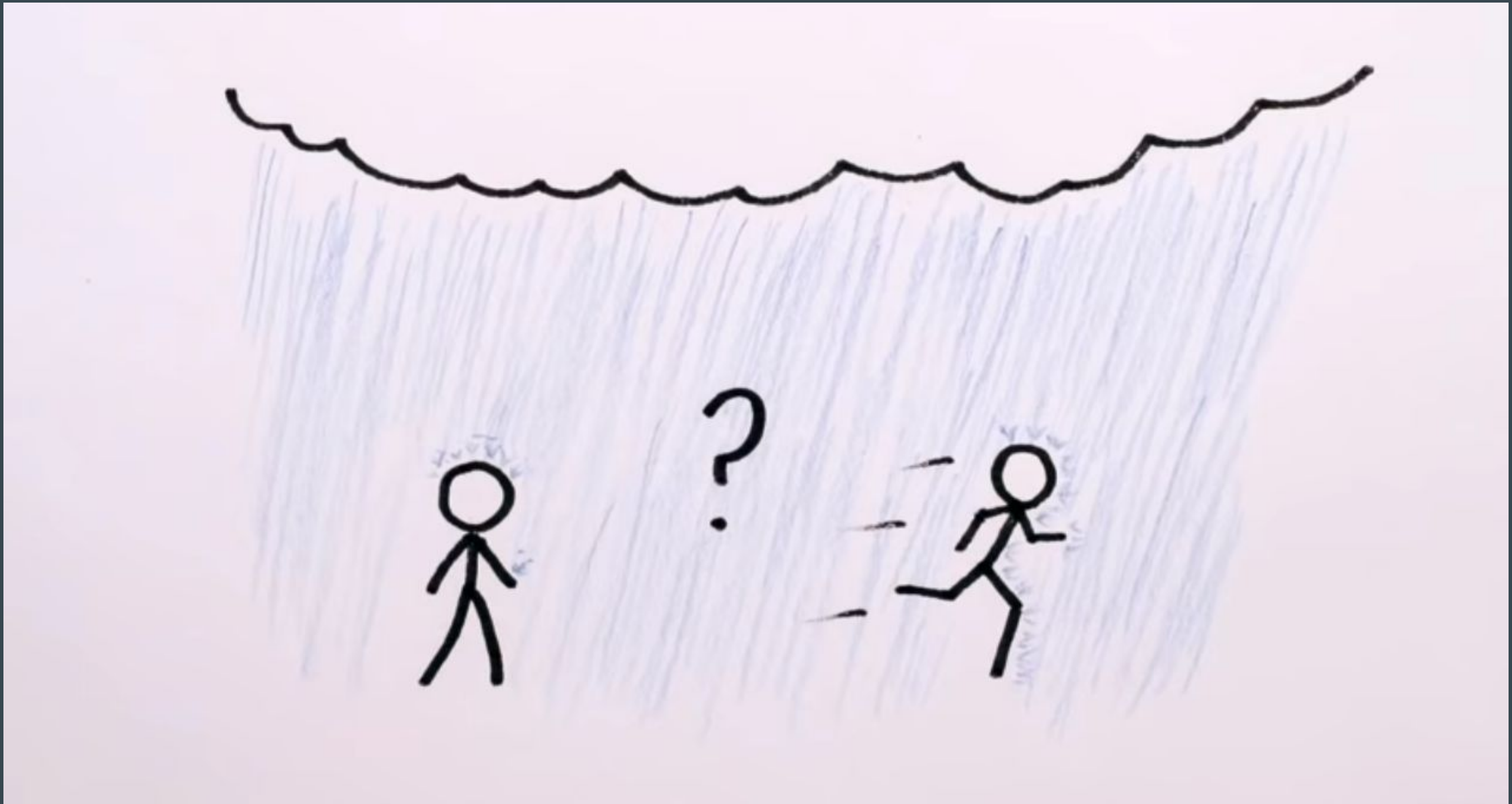
ISC 3313: Final Project Presentation

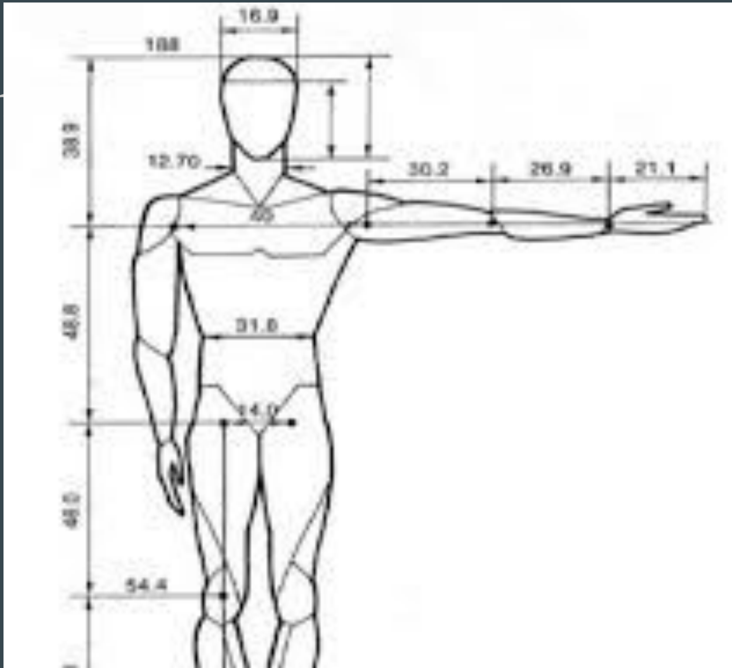
...

"Is it better to walk or run in the rain?"
Coded in C++



Background Research

- Anthropometrics: The study of the measurements of the human body.
  - Average human height: 1.77 meters
  - Average human boardness: .4 meters
  - Average human depth: .26 meters
- How many raindrops hit the ground per second or
  - Light rain: 151 drops per square meter per second
  - Moderate rain: 495 drops per square meter per second
  - Heavy rain: 818 drops per square meter per second
- How fast do humans run or walk?
  - Average human running speed: 6.7 meters per second
  - Average human walking speed: 1.4 meters per second
- How fast do raindrops fall?
  - Terminal velocity for raindrops is 10 meters per second.



Our key players: the private variables

```
class rainProblem {
    int wetness = 0;
    float time;
    //making a box
    float refresh = .1;
    float p1[3];
    float p2[3];
    float p3[3];
    float p4[3];
    //average male height 1.77m
    float height;
    //average male depth .25m
    float depth = .26;
    //average broadness .40 m
    float broadness = .4;
    //raindrops per second
    //drizzling = 15100 rps, moderate rain = 49500, heavy rain 81800 rps
    int rps;
    int raindrops = rps * time;
    vector< vector<float> > rd;
```

- int wetness = the integer our simulation function will return, the amount of raindrops that hit our human-sized box.
- float time = the time it takes for the box to walk/run from point A to B.
- float refresh = the amount of time in between simulations of the box moving
- float p#[3] = arrays that contain the x, y, and z position of the box
- float height, depth, broadness = contain the average measurements of humans
- int rps = the amount of raindrops that fall from the sky every second.
- int raindrops = the amount of raindrops we will use in our simulation. Product of rps and time.

```
int simulation(float rwspeed, int rainper, float h) {
    time = makeTime(rwspeed);
    height = h;
```

```
int simulation(float rwspeed, int rainper, float h) {
    time = makeTime(rwspeed);
    height = h;
    rps = rainper;
    p1[0] = 0; p1[1] = 0; p1[2] = .5 + (broadness/2);
    p2[0] = depth; p2[1] = 0; p2[2] = .5 - (broadness/2);
    p3[0] = 0; p3[1] = height; p3[2] = .5 + (broadness/2);
    p4[0] = depth; p4[1] = height; p4[2] = .5 - (broadness/2);
    wetness = 0;
    raindrops = rps * time;
    for (int i = 0; i < raindrops; i++) {
        makeitRain(i, time);
    }
    //terminal velocity for raindrops = 10 m/sD
    for (float i = 0; i <= time; i += refresh) {
        movebox(rwspeed);
        rainCheck(i, time);
        int k = i * 10;
        if (k % 10 == 0) {
            cout << "percentage time: " << (i / time) * 100 << endl;
        }
    }
    return wetness;
}
```

Step one: initialize all private variables to their starting positions and private variables simulation arguments.

Step two: make raindrops

Making it rain

```
void makeitRain(int i, float time) {
    //raindrop, droptop, smoking on cookie in a hotbox
    int tInt = time;
    vector<float> raindrop;
    raindrop.push_back((rand() % 100) + ((float)rand() / RAND_MAX));
    raindrop.push_back(10);
    raindrop.push_back((rand() % tInt) + ((float)rand() / RAND_MAX));
    raindrop.push_back(((float)rand() / RAND_MAX));
    rd.push_back(raindrop);
}
```

1. Cast time as an int
2. Create smaller raindrop vector
3. Give raindrop random x position between 0 and 100
4. Give raindrop y position 10 (arbitrary)
5. Give raindrop random trigger time between 0 and time t
6. Give raindrop random z position between 0 and 1
7. Push raindrop vector into vector rd

When we do this inside of a for loop, we create all of our raindrops!

```
int simulation(float rwspeed, int rainper, float h) {
    time = makeTime(rwspeed);
    height = h;
    rps = rainper;
    p1[0] = 0; p1[1] = 0; p1[2] = .5 + (broadness/2);
    p2[0] = depth; p2[1] = 0; p2[2] = .5 - (broadness/2);
    p3[0] = 0; p3[1] = height; p3[2] = .5 + (broadness/2);
    p4[0] = depth; p4[1] = height; p4[2] = .5 - (broadness/2);
    wetness = 0;
    raindrops = rps * time;
    for (int i = 0; i < raindrops; i++) {
        makeitRain(i, time);
    }
    //terminal velocity for raindrops = 10 m/sD
    for (float i = 0; i <= time; i += refresh) {
        movebox(rwspeed);
        rainCheck(i, time);
        int k = i * 10;
        if (k % 10 == 0) {
            cout << "percentage time: " << (i / time) * 100 << endl;
        }
    }
    return wetness;
}
```

Step one: initialize all private variables to their starting positions and private variables simulation arguments.

Step two: make raindrops

Step three: run our simulation inside of our for loop until we reach time t

Moving the box

```
void movebox(float rwspeed) {
    p1[0] += refresh*rwspeed;

    p2[0] = p1[0] + depth;

    p3[0] += refresh*rwspeed;

    p4[0] = p3[0] + depth;
}
```

Increase p1 and p3 x position by refresh rate * speed.
Increase p2 and p4 x position by p1 or p3 + side length.

Making rainfall and collision detection

```
void rainCheck(float i, float time) {
    raindrops = rd.size();
    for (int j = 0; j < raindrops; j++) {
        if (fabs(rd[j][2] - i) <= .001) {
            rd[j][1] -= 10 * refresh;
            rd[j][2] += refresh;
        }
        if (rd[j][0] >= p1[0] && rd[j][0] <= p2[0] &&
            rd[j][1] >= p1[1] && rd[j][1] <= p4[1] &&
            rd[j][3] <= p1[2] && rd[j][3] >= p4[2]) {
            wetness++;
            rd[j][1] == -1;
        }
    }
}
```

1. Set raindrop number to number of vectors in rd
2. Loop through the raindrops, if the raindrop' s trigger time is close ot the trigger time, move the raindrop' s x position by the terminal velocity of the drop * refresh.
3. Increase the raindrop' s trigger time by refresh so it keeps up with time.
4. Test for x,y,z collision between the raindrop and the box, if all conditions are satisfied, increase wetness by one and change raindrop' s y position to -1.

```
int simulation(float rwspeed, int rainper, float h) {
    time = makeTime(rwspeed);
    height = h;
    rps = rainper;
    p1[0] = 0; p1[1] = 0; p1[2] = .5 + (broadness/2);
    p2[0] = depth; p2[1] = 0; p2[2] = .5 - (broadness/2);
    p3[0] = 0; p3[1] = height; p3[2] = .5 + (broadness/2);
    p4[0] = depth; p4[1] = height; p4[2] = .5 - (broadness/2);
    wetness = 0;
    raindrops = rps * time;
    for (int i = 0; i < raindrops; i++) {
        makeitRain(i, time);
    }
    //terminal velocity for raindrops = 10 m/sD
    for (float i = 0; i <= time; i += refresh) {
        movebox(rwspeed);
        rainCheck(i, time);
        int k = i * 10;
        if (k % 10 == 0) {
            cout << "percentage time: " << (i / time) * 100 << endl;
        }
    }
    return wetness;
}
```

Step one: initialize all private variables to their starting positions and private variables simulation arguments.

Step two: make raindrops

Step three: run our simulation inside of our for loop until we reach time t

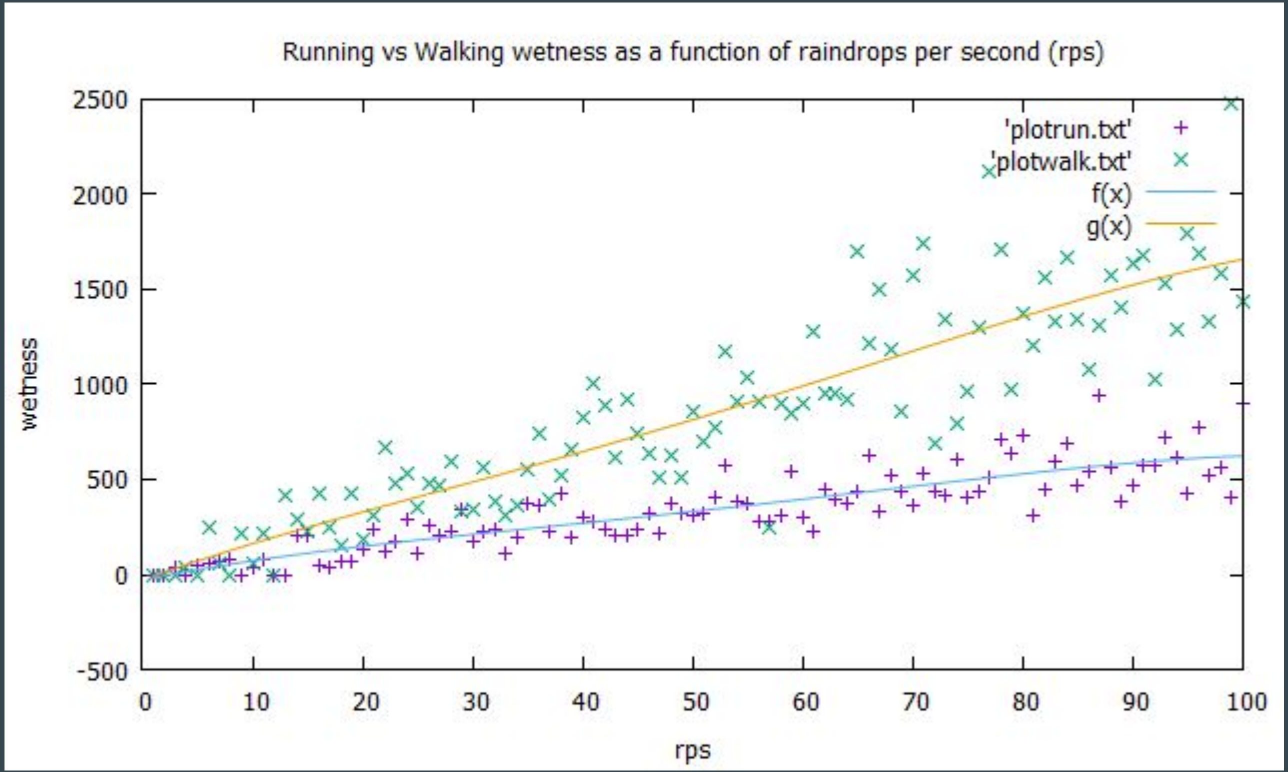
Step four: return final wetness after simulation finishes.

int main ()

```cpp
int main() {
    rainProblem walk;
    rainProblem run;
    //average human walking speed is 1.4m/s
    //average human running speed is 6.7m/s
    float runspeed = 6.7;
    float walkspeed = 2.8;
    //cout << run.simulation(runspeed, 1000, 1.77);
    int countr = 0;
    ofstream myfile("plotrun.txt");
    myfile << "100" << "\n";
    myfile << "x y" << "\n";
    for (int i = 15100; i <= 81800; i += 667) {
        cout << countr << "% complete " << endl;
        myfile << i << " " << run.simulation(runspeed,i,1.77) << "\n";
        countr++;
    }
    int countw = 0;
    ofstream myfile1("plotwalk.txt");
    myfile << "100" << "\n";
    myfile << "x y" << "\n";
    for (int i = 15100; i <= 81800; i += 6670) {
        cout << countw << "% complete " << endl;
        myfile1 << i << " " << walk.simulation(walkspeed, i, 1.77) << "\n";
        countw++;
    }
    return 0;
}
```

1. Declare objects to the rainProblem class.
2. Create run and walkspeed variables.
3. Write the calculations of the runspeed simulation into a file using a for loop.
    3.1 This for loop increases the raindrops per second of the simulation.
4. Write the calculations of the walkspeed simulation into a file using a for loop.
    4.1 This for loop also increases the raindrops per second of the simulation.
5. Return 0 and we're done! :)

Our results!



Running vs Walking wetness as a function of raindrops per second (rps)
2500
2000
1500
1000
500
0
-500
wetness
0
10
20
30
40
50
60
70
80
90
100
rps
'plotrun.txt'
'plotwalk.txt'
f(x)
g(x)