Disaster Simulator

New Mexico

Supercomputing Challenge

Final Report

April 4, 2018

THS116

Taos High School

Jenay Barela

Kineo Memmer

Jeremiah Martinez

Zak Willis

Teacher: Tracy Galligan

Project Mentor: Tracy Galligan

Summary:

This summer and in past years, natural disasters have caused billions of dollars of damage to people's homes, neighborhoods, and towns. Hurricane Harvey alone caused up to $180 billion of damage to residential areas, more than both Hurricanes Katrina and Sandy, which devastated New Orleans in 2005 and New York City in 2012, respectively. If there had been better planning in relation to the structural design of the buildings that were devastated, would less damage have occurred? The purpose of this project is to build a computer simulation that can accurately model different building types and show which aspects could be improved. The goal is to create a disaster simulator that can be used to help contractors and engineers design buildings to hold up better against natural disasters by using the parameters of disaster scales and building off of former knowledge from past disasters. The experimenters will use an integrated development environment and simulation coding to create this model. In addition, the experimenters will conduct research to determine which aspects of a building holds up the best against each respective natural disaster; hurricanes, earthquakes, and tornados.

Problem:

Can a computer model effectively demonstrate the effects of natural disasters on structures?

Method:

When first writing the code we had to decide what the model would look like and what it would do. Once we had come up with the design we started by writing the pseudo code to map out what code was necessary. The next step was creating the main page of the simulation. We started by adding buttons and labels. We then had to organize them on the screen. Finally we added code to the buttons so that they would switch pages. The next page we worked on was the

disaster selection page. On this page we added three buttons to the screen which would allow the user to choose which disaster they wanted to test their building on. We later removed this page and added it as a drop down option. The reason we decided to switch from having a separate page for the disaster choice came as we were making the pre built model page.

Once we had placed the models on the screen and prepared them for testing we realized that it was inefficient to have a separate page for choosing the disasters. The separate page added unnecessary cons to the project. We decided that on the pre built building page we would have the building the user chooses to come with a pre chosen disaster. Once the prebuilt building page was finished we started the code to test the building against the different disasters. In order to do this we decided on a base damage for the disasters. We then subtracted the base damage from the total strength of the building and had the final value display the screen. Once we had finished the coding of the different pre built building and the display screen we ran into another problem involving how best to get the model to come up with a total for the building when the user builds their own building. There are over two million possibilities for buildings in our model and we had to come up with an efficient way of making all of the different possibilities possible. The way we accomplished this is not the most efficient but we plan to improve the efficiency.

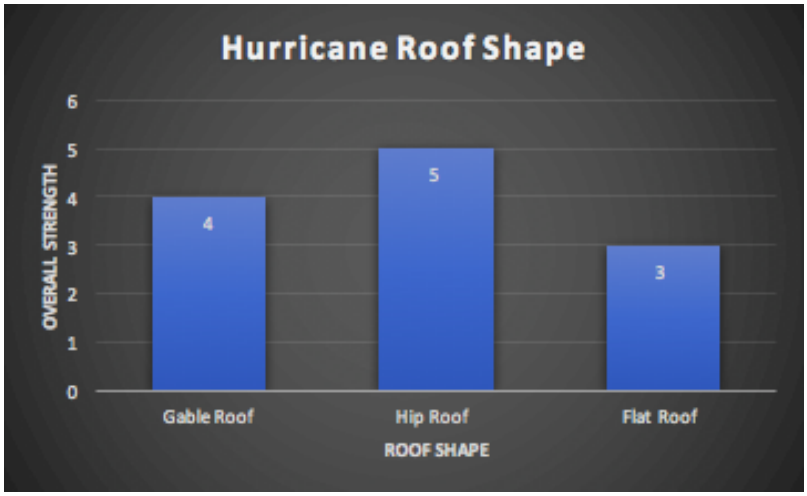
Once it was possible to test all the different possibilities all that was left to code was the end page that displays the status of the building. On the end page the code displays the total of the building before the disaster and the total after. There is also a scale on the page that gives the user an idea of how their building held up. Once we had finished the code we ran tests to make sure the data we were outputting was correct. We did this by comparing the research to the output of the model.

In addition to writing the code, we conducted research to determine which combination of aspects would hold up best against each respective natural disaster. First, we created a list of aspects that are impacted the most by tornadoes, hurricanes, and earthquakes. We decided on twelve separate aspects: foundation, windows, exterior walls, roofing, doors, garage doors, roof shape, flooring, ceiling and wall support, framing, height, and soil around the building. In addition, some of the aspects do not impact the strength of a building against certain disasters. For example, type of flooring does not impact tornadoes, and soils type does not impact hurricanes.

After choosing the aspects, we did extensive research to create a list of materials for each aspect. For example, our code allows you to choose between five different foundation types: crawl space, basement, slab, pile and girder, and crawl space, basement, or slab with retrofitting. The next step in our research was to determine which material from each list allegedly holds up the best against each respective natural disaster. To find out this information, we searched through architecture websites, watched videos of natural disasters online, and read through reviews and suggestions for building homes. We primarily conducted the research online, but talked to some local sources to get more information. After conducting this research, we were able to put together a comprehensive list and assign a number scale to rate each material aspect for each disaster that was later applied into the computer program.

Discussion:

To verify our model, we ran multiple trials and graphed the results to see if the code matched our research and produced the results that we expected. For the most part, the model produced the results that we expected. Here is an example:



We ran the code with a set of constants and changed only the roof shape variable, recording the durability of that house according to the parameters that we set in the code. From the graph you can see that a hip roof holds up the best against a hurricane, with gable coming in second and a flat roof being the worst. This matches our research and is what we expected the results to be. A hip roof holds up better than a flat roof because flat roofs have higher forces on the corners and eves that act like sails in strong winds, allowing a flat roof to be lifted off the house in the strong winds that hurricanes produce. On the other hand, hip roofs slope in four directions, withstanding winds better. In addition, hip roofs have more surface area, allowing wind pressure to equalize better than it does on flat or gable roofs (1). Overall, the results of our test runs mostly matched the information we gathered in our research, which verifies our model.

Results:

The results of our study are shown in the results of running the code, as shown above. Our program allows users to choose their own material aspects used to build a home, and test it against one of the three natural disasters. The program then generates a number based off of each individual aspect's strength and the damage inflicted by the natural disaster. A number is then generated, and a scale is provided to compare your resultling number to and better understand

how well your house would hold up against the disaster you chose. The users can also choose from any of our nine building models. These models show which combination of material aspects would hold up best, medium, and worst against each natural disaster. These models are both a good reference point for the user, and represent the best combination of aspects for a home (as determined by our research) for each disaster. The hope is that homeowners, contractors, and engineers will use our model to help them build safer homes in areas that suffer heavy damage from natural disasters.

Conclusion:

Based on the results of our study we determined that a model can effectively demonstrate the effect of hurricanes, tornadoes, and earthquakes on structures. The model provides a wide range of options, allowing the user to create the strongest structure. In the future we would like to have a visual representation to go along with our model to show how the disaster affected the house designed. As well as expand the options for the different types of building materials to allow for more options when designing the houses. We would also like to make it possible for the user to test the premade houses against other types of disasters. We will continue to optimize the code to make it as efficient as possible. We would also like to provide suggestions to the user after they test their building as to how they could improve the strength of their building. We eventually would like to patent our software and offer it for commercial use.

Data:

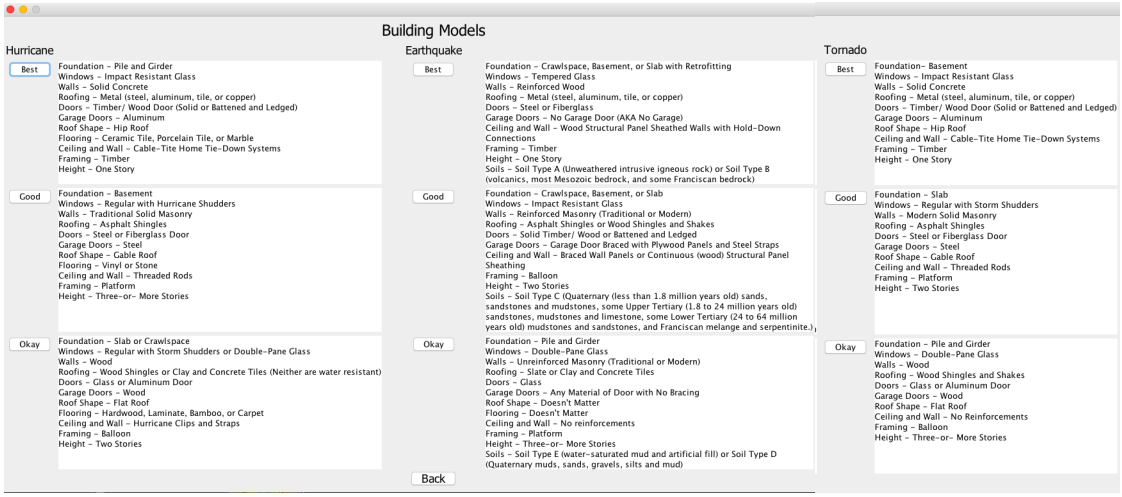
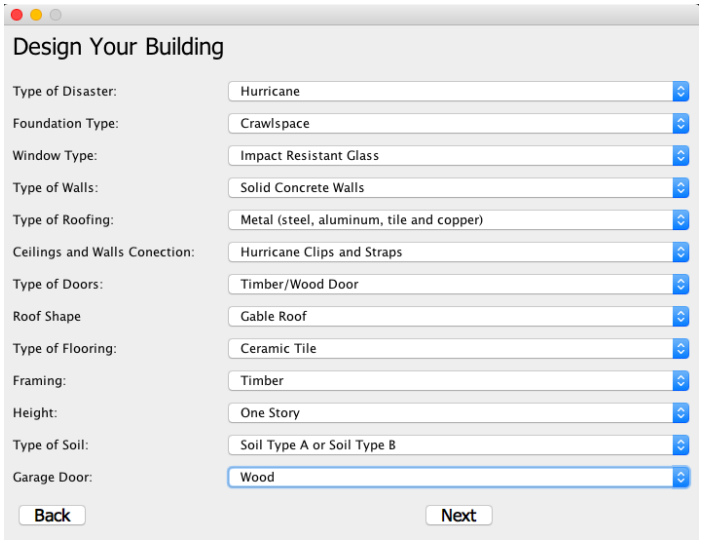


Fig. 1 and Fig 2. This is the building models page of our code, which displays the best, middle, and worst building combinations for each natural disaster. This page gives you the option to choose one of these pre-designed buildings and see how it holds up against each natural disaster. These buildings are also a reference point from which the user can design their own building.



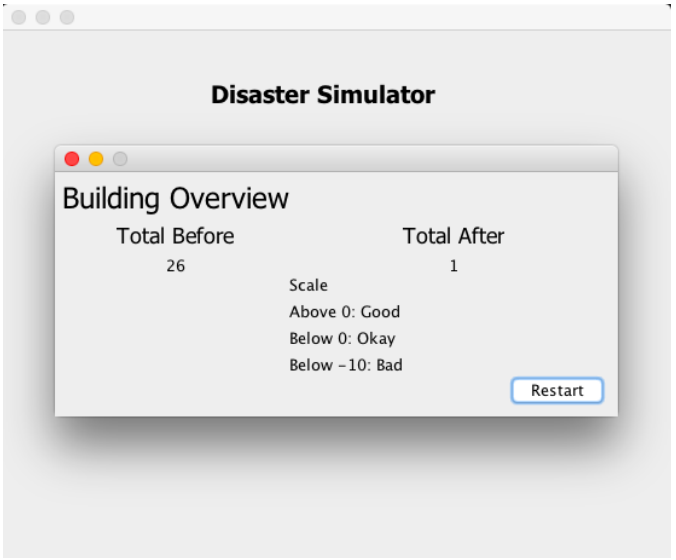


Fig. 3 and Fig. 4. These images show the design your own building page. On this page, you can choose which natural disaster to test your building against, what foundation type you want, what window type you want, etc. After you are done choosing your desired aspects, click the “Next”

button, which takes you to the building overview page, as shown in Fig. 4, which gives you a total before the disaster, a total after damage is inflicted, and a scale to compare your results to.

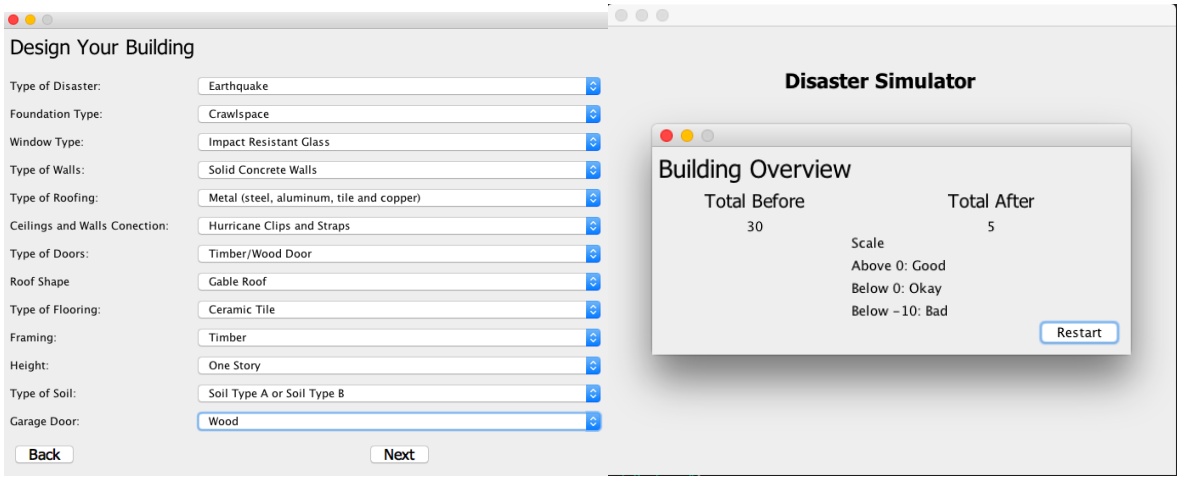


Fig. 5 and Fig. 6. This is another example of a possible combination of building aspects, tested against an earthquake (the building in Fig. 4 and Fig. 5 were tested against a hurricane). As shown in Fig. 6, this building gets a score of five after damage from the earthquake is inflicted, which is above zero and good according to the scale.

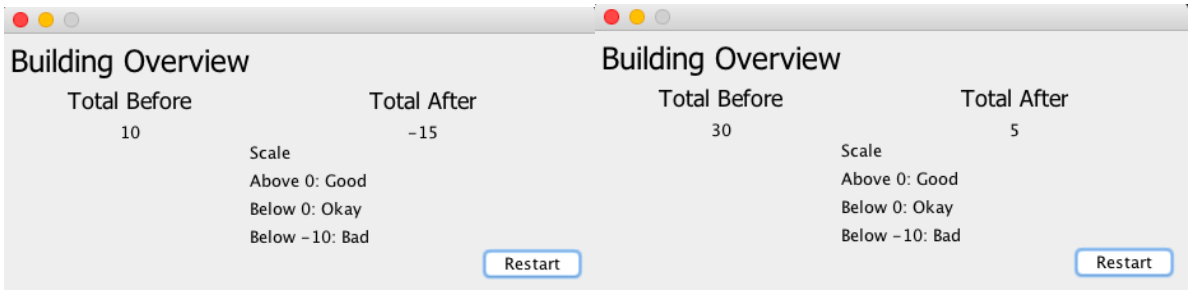
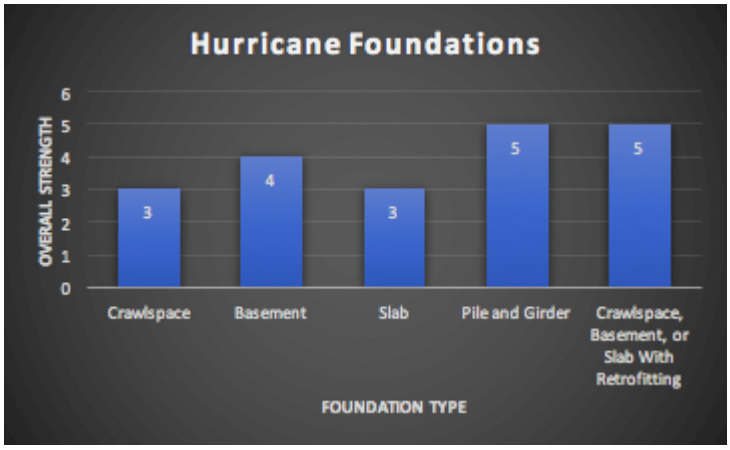
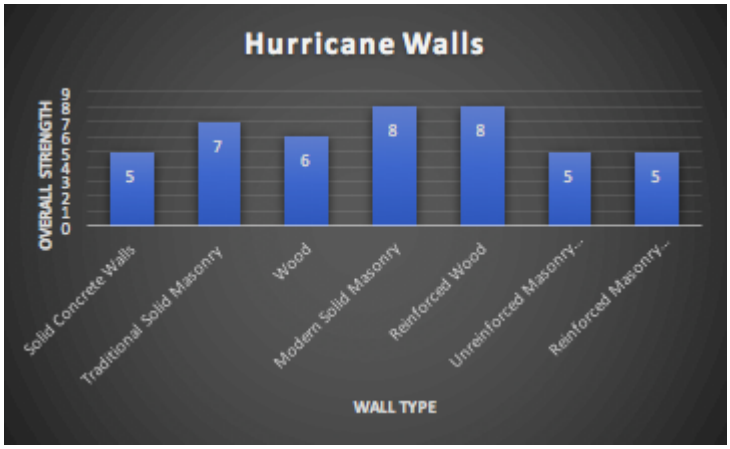


Fig. 7 and Fig. 8. These are two more examples of the building overview page and possible scores for buildings that users created. As shown in the images, both disasters inflicted twenty-five points of damage. The building in Fig. 8 is better, because it started with a total of thirty, and ended with a total of five. The building in Fig. 7 was not as good, because it only started with a total of ten, so it ended with negative fifteen.



Foundation Type	Overall Strength
Crawlspace	3
Basement	4
Slab	3
Pile and Girder	5
Crawlspace, Basement, or Slab With Retrofitting	5

Fig. 9 and Table 1. This is another example of the results of running multiple trials to verify our model. In this set of trials, we tested to see which foundation type would hold up the best against a hurricane. We chose a set of constants for the rest of the aspects, and ran the simulation with a different foundation type every time. As shown in Fig. 9, pile and girder and crawl space, basement, or slab with retrofitting both hold up the best against a hurricane. This conclusion also matches our research.



Wall Type	Overall Strength
Solid Concrete Walls	5
Traditional Solid Masonry	7
Wood	6
Modern Solid Masonry	8
Reinforced Wood	8
Unreinforced Masonry (Traditional or Modern)	5
Reinforced Masonry (Traditional or Modern)	5

Fig. 10 and Table 2. This is another example of the results of running multiple trials to verify our model. In this set of trials, we tested to see which wall type would hold up the best against a hurricane. We chose a set of constants for the rest of the aspects, and ran the simulation with a different wall type every time. As shown in Fig. 10, modern solid masonry and reinforced wood walls both hold up the best against a hurricane. This conclusion also matches our research.

Achievements:

We were able to successfully model the effects of natural disasters on structures. We accomplished this by working together as a team and relying on each other to help solve and improve the code for our simulation. Another accomplishment we have is that this model is one of the only computer models to show the effects of natural disasters effects on buildings. When we were first starting this project we looked into what models already existed based around this idea. From our research we found a limited number of computer models existed to show the effects of natural disasters on buildings.

Acknowledgements:

Mrs. Galligan- Sponsor of supercomputing at Taos High School. Provided a workspace after school and on the weekends.

Sources:

BuildingMuseum. “Which Roof Shape Best Stands up to the Fiercest Gale?” *YouTube*, YouTube, 16 May 2014, www.youtube.com/watch?v=iIWfT9HpqnI.

“Tornado Classification The F-Scale: Wind Speed and Damage.” *Tornado Classification, F-Scale*, www.enchantedlearning.com/subjects/weather/tornado/fscale.shtml.

“Saffir-Simpson Hurricane Wind Scale.” *Saffir-Simpson Hurricane Wind Scale*, National Oceanic and Atmospheric Administration, www.nhc.noaa.gov/aboutsshws.php.

“Earthquake Retrofitting: House Bolting, Foundation Bolting & Cripple Wall Bracing.”*Earthquake Safety*, 2014, www.geo.mtu.edu/UPSeis/magnitude.html.

Sun-Sentinel, South Florida. “Hurricane Shutter Guide: Compare Types.” *Sun-Sentinel.com*, 6 Sept. 2017, www.sun-sentinel.com/news/weather/hurricane/sfl-hc-shutterguide-htmlstory.html.

Mishra, Gopal. “Performance of Various Types of Buildings during Earthquake.” *The Constructor*, 17 Sept. 2017, theconstructor.org/earthquake/performance-buildings-types-earthquake/2224/.

Pollack, Andrew. “The World; One Defense Against Quakes: Build Homes of Wood.” *The New York Times*, 31 Jan. 1999, www.nytimes.com/1999/01/31/weekinreview/the-world-one-defense-against-quakes-build-homes-of-wood.html.

“WOOD LIGHT-FRAME CONSTRUCTION .” FEMA.

Whitworth Builders, Inc. “The Best Hurricane Proof Construction Materials.” *Whitworth Builders*, 8 Nov. 2017,

whitworthbuilders.com/what-construction-material-is-the-best-for-a-hurricane-proof-hou e/.

Timmons, Jessica. “Your Sourcefor Metal Roofing.” *The Best Roof for High Winds*, www.metalroofnet.com/metal-roofing-blog/bid/67114/The-Best-Roof-for-High-Winds.

Team, Whirlwind. “Are Metal Roofs Wind Resistant?” *Metal Buildings*, Whirlwind Team, 2 July 2015, www.whirlwindsteel.com/blog/bid/406259/are-metal-roofs-wind-resistant.

PJ's Roofing. “Blog.” *PJsRoofing*, 8 Feb. 2017, www.roofingbypj.com/best-home-roofing-materials-for-high-wind-areas/.

All-Nu Construction. “Which Roofing Materials Stand Up Best to Damage?” *All-Nu Construction.* , 4 Aug. 2016, all-nuconstruction.com/which-roofing-materials-stand-up-best-to-damage/.

Federal Alliance for Safe Homes, Inc. “Tornadoes: Roof - Choosing the Right Covering.”*FLASH*, 2018, flash.org/peril_inside.php?id=176.

Hurricane Proof. “Roofs.” *Roofs : Hurricane Protection : Damage Prevention*, 2018, www.hurricaneproof.com/roofing.php.

“Case Studies.” *NZ Wood*,m www.nzwood.co.nz/uncategorized/why-do-some-buildings-survive-earthquakes-when-o hers-fail/.

“The Weightiness of Roofing.” *Classic Metal and Roofing Systmes*, 6 May 2014, www.classicmetalroofingsystems.com/2014/05/06/the-weightiness-of-roofing/

FDEM. *FloridaDisaster.org*, FDEM, 2018, www.floridadisaster.org/Response/engineers/HES/Manual/16--Appendix%20E%20-%2 Roofing%20Material%20Weights.pdf.

Chen, Chia-chi, et al. “EARTHQUAKE DAMAGE OF WINDOW GLASS AND DOOR OF A HOTEL DAMAGED IN THE 1999 CHI-CHI TAIWAN EARTHQUAKE.” *13th World Conference on Earthquake Engineering*, www.iitk.ac.in/nicee/wcee/article/13_1444.pdf.

“How to Improve Your Home's Earthquake Resistance.” *Chelsea Green Publishing*, 2 Feb. 2009, www.chelseagreen.com/blogs/wtf-earthquake-resistance-draft/.

“ .” *Living with Earthquakes in the Pacific Northwest*, oregonstate.edu/instruct/oer/earthquake/13 chapter 11_color.html.

Joyce, Christopher. “Wood-Frame House Suffers in Simulated Earthquake.” *NPR*, NPR, 3 Jan. 2007, www.npr.org/templates/story/story.php?storyId=6716119.

“Flooring for Flood-Prone Areas.” *FlooringInc Blog*, 7 Dec. 2017, www.flooringinc.com/blog/flooring-for-flood-prone-areas/.

Hicks, Angie. “Choosing Flooring for Rooms That Get Wet.” *Angie's List | Join for FREE to See 10 Million Verified Reviews*, 29 Apr. 2016, www.angieslist.com/articles/choosing-flooring-rooms-get-wet.htm.

FEMA. “Chapter 5: WALLS.” *Homebuilders' Guide to Earthquake-Resistant Design and Construction*,www.fema.gov/media-library-data/20130726-1535-20490-9409/fema232pt2.pf.

“Tornado Myths and Facts.” *Tornado Myths and Facts*, readywisconsin.wi.gov/tornado/tornado_myths_facts.asp.

Fessenden, Ford, et al. “Water Damage From Hurricane Harvey Extended Far Beyond Flood Zones.” *The New York Times*, The New York Times, 2 Sept. 2017, www.nytimes.com/interactive/2017/09/01/us/houston-damaged-buildings-in-fema-flood-zones.html.

“Soil Type and Shaking Hazard in the San Francisco Bay Area.” *U.S. Geological Survey*, earthquake.usgs.gov/hazards/urban/sfbay/soiltype/.

“Designing Earthquake Safe Buildings and Structures.” *StruCalc*, www.strucalc.com/designing-earthquake-safe-buildings-and-structures/.

“The Best Hurricane Proof Construction Materials.” *Whitworth Builders*, 8 Nov. 2017, whitworthbuilders.com/what-construction-material-is-the-best-for-a-hurricane-proof-house/.

Gibbs, Tony. “Hurricanes and Their Effects on Buildings and Structures in the Caribbean.”*Hurricanes and Their Effects on Buildings and Structures in the Caribbean*, 2001, www.oas.org/pgdm/document/bitc/papers/gibbs/gibbs_01.htm.

“Flooring for Flood-Prone Areas.” *FlooringInc Blog*, 7 Dec. 2017, www.flooringinc.com/blog/flooring-for-flood-prone-areas/.

Fessenden, Ford, et al. “Water Damage From Hurricane Harvey Extended Far Beyond Flood Zones.” *The New York Times*, The New York Times, 2 Sept. 2017, www.nytimes.com/interactive/2017/09/01/us/houston-damaged-buildings-in-fema-flood-zones.html.

“Hurricane Resistant Homes | Wind Resistant.” *Deltec Homes*, www.deltechomes.com/learn-more/hurricane-resistance/.

DeMatto, Amanda. “8 Ways to Protect Your Home Against Tornadoes and Hurricanes.”*Popular*

Mechanics, Popular Mechanics, 15 Feb. 2018, www.popularmechanics.com/home/interior-projects/how-to/g605/8-ways-to-protect-you-home-against-tornadoes-and-hurricanes/.

"Types of Roofing Material." Thehousedesigners.com, www.thehousedesigners.com/articles/roofing-material.asp.

"How Earthquakes Affect Buildings." Commercial Interior Design Excellence, www.interiorsandsources.com/article-details/articleid/9383/title/how-earthquakes-affe.

"Disaster Resistance ." Disaster Resistance - Disaster Resistant Buildings - Durable Building, 2018, www.concretethinker.com/solutions/Disaster-Resistance.aspx.

Fritsch, Jane. "THE EARTHQUAKE: Why Buildings Collapsed; Quake Revealed Flaws in 'Safe' Structures' Design." The New York Times, The New York Times, 20 Jan. 1994, www.nytimes.com/1994/01/20/us/earthquake-why-buildings-collapsed-quake-revealed-faws-safe-structures-design.html.

"Hurricane Resistant Homes | Wind Resistant." Deltec Homes, Deltec Homes, www.deltechomes.com/learn-more/hurricane-resistance/.

SeniorQuack. "Stronger Homes After Katrina." YouTube, YouTube, 12 Dec. 2012, www.youtube.com/watch?annotation_id=annotation_436043&feature=iv&src_vid=sT7z8LYlx8&v=jP7WViY3crA.

SeniorQuack. "Huricane-Proof Houses." YouTube, YouTube, 13 Dec. 2012, www.youtube.com/watch?annotation_id=annotation_688141&feature=iv&src_vid=jP7ViY3crA&v=sTM7z8LYlx8.

PreciseForms. "Building Hurricane, Tornado, and Storm Resistant Concrete Homes Using

Aluminum Concrete Forms." YouTube, YouTube, 15 Aug. 2012, www.youtube.com/watch?v=fiBmRzIayaI.

TheWeatherChannel. "Does Concrete Protect from a Hurricane?" YouTube, YouTube, 8 Aug. 2013, www.youtube.com/watch?v=LAMVgQOKqIU.

Storm. "The 'Dome Home' Fends Off Hurricanes." YouTube, YouTube, 29 Aug. 2012, www.youtube.com/watch?v=jxsSBHTFk3w.

dphanlon4. "Hurricane Proof Home, 6 Minutes." YouTube, YouTube, 30 Apr. 2012, www.youtube.com/watch?v=p89K8uYNsJI.

"BUILDING BACK STRONGER KEEPS NEIGHBORHOODS ALIVE ." FEMA.

"Resources for Reconstruction after 2016 Louisiana Flooding ." FEMA, Sept. 2016.

PinoyHowTo. "How to Build an Earthquake Proof and Typhoon Proof House | PinoyHowTo." YouTube, YouTube, 28 Sept. 2015, www.youtube.com/watch?v=3DuGhgIl22s.

gmanews. "SONA: 'Waffle Box House', Kaya Raw Labanan Ang Anumang Lakas Ng Bagyo at Lindol." YouTube, YouTube, 2 June 2015, www.youtube.com/watch?v=tbvBDfEg0rc.

Harris, William. "How Earthquake-Resistant Buildings Work." HowStuffWorks Science, HowStuffWorks, 8 Mar. 2018, science.howstuffworks.com/engineering/structural/earthquake-resistant-buildings1.htm.

"Garage Cube Underground Storm Shelter." Survive-a-Storm Shelters, survive-a-storm.com/product/garage-cube-underground-storm-shelter/.

DeMatto, Amanda. "8 Ways to Protect Your Home Against Tornadoes and Hurricanes." Popular Mechanics, Popular Mechanics, 15 Feb. 2018,

www.popularmechanics.com/home/interior-projects/how-to/g605/8-ways-to-protect-your-home-against-tornadoes-and-hurricanes/.

Nosowitz, Dan. “Can You Tornado-Proof A Home?” Popular Science, Popular Science, 31 May 2013, www.popsci.com/technology/article/2013-05/can-you-tornado-proof-home.

Gibbs, Tony. “Hurricanes and Their Effects on Buildings and Structures in the Caribbean.” Hurricanes and Their Effects on Buildings and Structures in the Caribbean, 22 Jan. 2001, www.oas.org/pgdm/document/bitc/papers/gibbs/gibbs_01.htm.

Code

BuildingModels.java

```
package Disaster.Simulator;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;


public class BuildingModels extends javax.swing.JFrame {
	public static int stotal;
	public static int type;
	public static int endTot;

	public BuildingModels() {
		createMenu();
	}

	private void createMenu() {

		setVisible(true);
		setDefaultCloseOperation(JFrame.EXIT_ON_CLOSE);

		JLabel Title = new JLabel("Building Models");
		JButton Back = new JButton("Back");

		JLabel hurricain = new JLabel("Hurricane");
	// Declares "Hurricane," "Best," "Good," and "Okay,"
		JButton hBest = new JButton();
		// buttons and their respective text panes
		JTextPane bestH = new JTextPane();
		JButton hGood = new JButton();
		JTextPane goodH = new JTextPane();
		JButton hOkay = new JButton();
		JTextPane okayH = new JTextPane();

		JLabel earthquake = new JLabel("Earthquake");
	// Declares "Earthquake," "Best," "Good," and "Okay,"
		JButton eBest = new JButton();
		// buttons and their respective text panes
```

```
        JTextPane bestE = new JTextPane();
        JButton eGood = new JButton();
        JTextPane goodE = new JTextPane();
        JButton eOkay = new JButton();
        JTextPane okayE = new JTextPane();

        JLabel tornado = new JLabel("Tornado");
    // Declares "Tornado," "Best," "Good," and "Okay,"
        JButton tBest = new JButton();
        // buttons and their respective text panes
        JTextPane bestT = new JTextPane();
        JButton tGood = new JButton();
        JTextPane goodT = new JTextPane();
        JButton tOkay = new JButton();
        JTextPane okayT = new JTextPane();

        Title.setFont(new Font("Tahoma", 0, 24));
        Back.setFont(new Font("Tahoma", 0, 18));
        Back.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                BackActionPreformed(evt);
            }

        });

        hBest.setText("Best");
        hBest.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                Besther(evt);
            }
        });
        bestH.setText("Foundation - Pile and Girder\r\n" +
                    // Adds the variables descriptions for the
                    "Windows - Impact Resistant Glass\r\n" +
                        // house that holds up best against hurricanes
                    "Walls - Solid Concrete\r\n" +
                                // into the BuildingModels page
                    "Roofing - Metal (steel, aluminum, tile, or copper)\r\n" +
                    "Doors - Timber/ Wood Door (Solid or Battened and Ledged)\r\n"
+
```

```
                        "Garage Doors - Aluminum\r\n" +
                        "Roof Shape - Hip Roof\r\n" +
                        "Flooring - Ceramic Tile, Porcelain Tile, or Marble\r\n" +
                        "Ceiling and Wall - Cable-Tite Home Tie-Down Systems\r\n" +
                        "Framing - Timber\r\n" +
                        "Height - One Story");
                hGood.setText("Good");
                hGood.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent evt) {
                                Goodher(evt);
                        }
                });
                goodH.setText("Foundation - Basement\r\n" +
                                        // Adds the variable descriptions for the
                        "Windows - Regular with Hurricane Shudders\r\n" +
                                        // house that holds up okay against hurricanes
                        "Walls - Traditional Solid Masonry\r\n" +
                                        // into the BuldingModels page
                        "Roofing - Asphalt Shingles\r\n" +
                        "Doors - Steel or Fiberglass Door\r\n" +
                        "Garage Doors - Steel\r\n" +
                        "Roof Shape - Gable Roof\r\n" +
                        "Flooring - Vinyl or Stone\r\n" +
                        "Ceiling and Wall - Threaded Rods\r\n" +
                        "Framing - Platform\r\n" +
                        "Height - Three-or- More Stories");
                hOkay.setText("Okay");
                hOkay.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent evt) {
                                Okayher(evt);
                        }
                });
                okayH.setText("Foundation - Slab or Crawlspace\r\n" +
                                                        // Adds the variable descriptions for
the
                        "Windows - Regular with Storm Shudders or Double-Pane
Glass\r\n" +                                            // house that holds up the worst
against
```

```
                        "Walls - Wood\r\n" +
                                                                    // hurricanes
into the BuildingModels page
                        "Roofing - Wood Shingles or Clay and Concrete Tiles (Neither are
water resistant)\r\n" +
                        "Doors - Glass or Aluminum Door\r\n" +
                        "Garage Doors - Wood\r\n" +
                        "Roof Shape - Flat Roof\r\n" +
                        "Flooring - Hardwood, Laminate, Bamboo, or Carpet\r\n" +
                        "Ceiling and Wall - Hurricane Clips and Straps\r\n" +
                        "Framing - Balloon\r\n" +
                        "Height - Two Stories");
            hurricain.setFont(new Font("Tahoma", 0, 18));

            eBest.setText("Best");
            eBest.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent evt) {
                    Bestert(evt);
                }
            });
            bestE.setText("Foundation - Crawlspace, Basement, or Slab with Retrofitting\r\n"
+                                   // Adds the variable descriptions for the
                        "Windows - Tempered Glass\r\n" +
                                                                    // house that holds up
the best against
                        "Walls - Reinforced Wood\r\n" +
                                                                    // earthquakes into the
BuildingModels page
                        "Roofing - Metal (steel, aluminum, tile, or copper)\r\n" +
                        "Doors - Steel or Fiberglass\r\n" +
                        "Garage Doors - No Garage Door (AKA No Garage)\r\n" +
                        "Ceiling and Wall - Wood Structural Panel Sheathed Walls with
Hold-Down\r\n" +
                        "Connections\r\n" +
                        "Framing - Timber\r\n" +
                        "Height - One Story\r\n" +
                        "Soils - Soil Type A (Unweathered intrusive igneous rock) or Soil
Type B\r\n" +
                        "(volcanics, most Mesozoic bedrock, and some Franciscan
bedrock)");
```

```
                eGood.setText("Good");
                eGood.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent evt) {
                                Goodert(evt);
                        }
                });
                goodE.setText("Foundation - Crawlspace, Basement, or Slab\r\n" +
                                                        // Adds the variable descriptions for the
                                "Windows - Impact Resistant Glass\r\n" +
                                                                // house that holds up okay
against
                                "Walls - Reinforced Masonry (Traditional or Modern)\r\n" +
                                                                // earthquakes into the
BuildingMOdels page
                                "Roofing - Asphalt Shingles or Wood Shingles and Shakes\r\n" +
                                "Doors - Solid Timber/ Wood or Battened and Ledged\r\n" +
                                "Garage Doors - Garage Door Braced with Plywood Panels and
Steel Straps\r\n" +
                                "Ceiling and Wall - Braced Wall Panels or Continuous (wood)
Structural Panel\r\n" +
                                "Sheathing\r\n" +
                                "Framing - Balloon\r\n" +
                                "Height - Two Stories\r\n" +
                                "Soils - Soil Type C (Quaternary (less than 1.8 million years old)
sands,\r\n" +
                                "sandstones and mudstones, some Upper Tertiary (1.8 to 24
million years old)\r\n" +
                                "sandstones, mudstones and limestone, some Lower Tertiary (24 to
64 million\r\n" +
                                "years old) mudstones and sandstones, and Franciscan melange
and serpentinite.)");
                eOkay.setText("Okay");
                eOkay.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent evt) {
                                Okayert(evt);
                        }
                });
                okayE.setText("Foundation - Pile and Girder\r\n" +
                                                                // Adds the variable descriptions for
the
```

```
                        "Windows - Double-Pane Glass\r\n" +
                                                                // house that holds up
the worst against
                        "Walls - Unreinforced Masonry (Traditional or Modern)\r\n" +
                                                // earthquakes into the BuildingModels page
                        "Roofing - Slate or Clay and Concrete Tiles\r\n" +
                        "Doors - Glass\r\n" +
                        "Garage Doors - Any Material of Door with No Bracing\r\n" +
                        "Roof Shape - Doesn't Matter\r\n" +
                        "Flooring - Doesn't Matter\r\n" +
                        "Ceiling and Wall - No reinforcements\r\n" +
                        "Framing - Platform\r\n" +
                        "Height - Three-or- More Stories\r\n" +
                        "Soils - Soil Type E (water-saturated mud and artificial fill) or Soil
Type D\r\n" +
                        "(Quaternary muds, sands, gravels, silts and mud)");
            earthquake.setFont(new Font("Tahoma", 0, 18));

            tBest.setText("Best");
            tBest.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent evt) {
                    BestTorn(evt);
                }
            });
            bestT.setText("Foundation- Basement\r\n" +
                                                                // Adds the variable
descriptions for the
                        "Windows - Impact Resistant Glass\r\n" +
                                                        // house that holds up the best
against
                        "Walls - Solid Concrete\r\n" +
                                                                // tornados
into the BuildingModels page
                        "Roofing - Metal (steel, aluminum, tile, or copper)\r\n" +
                        "Doors - Timber/ Wood Door (Solid or Battened and Ledged)\r\n"
+
                        "Garage Doors - Aluminum\r\n" +
                        "Roof Shape - Hip Roof\r\n" +
                        "Ceiling and Wall - Cable-Tite Home Tie-Down Systems\r\n" +
                        "Framing - Timber\r\n" +
```

```
                "Height - One Story");
        tGood.setText("Good");
        tGood.addActionListener(new java.awt.event.ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                GoodTorn(evt);
            }
        });
        goodT.setText("Foundation - Slab\r\n" +
                                                    // Adds the variable
descriptions for the
                "Windows - Regular with Storm Shudders\r\n" +
                                                    // house that holds up okay
against
                "Walls - Modern Solid Masonry\r\n" +
                                                        // tornados into the
BuildingModels page
                "Roofing - Asphalt Shingles\r\n" +
                "Doors - Steel or Fiberglass Door\r\n" +
                "Garage Doors - Steel\r\n" +
                "Roof Shape - Gable Roof\r\n" +
                "Ceiling and Wall - Threaded Rods\r\n" +
                "Framing - Platform\r\n" +
                "Height - Two Stories");
        tOkay.setText("Okay");

        tOkay.addActionListener(new ActionListener() {
            public void actionPerformed(ActionEvent evt) {
                okayTorn(evt);
            }
        });


        okayT.setText("Foundation - Pile and Girder\r\n" +
                                                // Adds the variable descriptions for
the
                "Windows - Double-Pane Glass\r\n" +
                                                        // house that holds up
the worst against
```

```
                                "Walls - Wood\r\n" +
                                                                        // tornados
into the BuildingModels page
                                "Roofing - Wood Shingles and Shakes\r\n" +
                                "Doors - Glass or Aluminum Door\r\n" +
                                "Garage Doors - Wood\r\n" +
                                "Roof Shape - Flat Roof\r\n" +
                                "Ceiling and Wall - No Reinforcements\r\n" +
                                "Framing - Balloon\r\n" +
                                "Height - Three-or- More Stories");
                tornado.setFont(new Font("Tahoma", 0, 18));

                // Sets the layout for the BuildingModels page
                GroupLayout layout = new GroupLayout(getContentPane());
                getContentPane().setLayout(layout);
                layout.setAutoCreateGaps(true);
                layout.setAutoCreateContainerGaps(true);
                layout.setHorizontalGroup(
                        layout.createSequentialGroup()

        .addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                        .addComponent(hurricain)
                                        .addComponent(hBest)
                                        .addComponent(hGood)
                                        .addComponent(hOkay))

        .addGroup(layout.createParallelGroup(GroupLayout.Alignment.TRAILING)
                                        .addComponent(bestH)
                                        .addComponent(goodH)
                                        .addComponent(okayH))

        .addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                        .addComponent(Title)
                                        .addComponent(earthquake)
                                        .addComponent(eBest)
                                        .addComponent(eGood)
                                        .addComponent(eOkay)
                                        .addComponent(Back))

        .addGroup(layout.createParallelGroup(GroupLayout.Alignment.TRAILING)
```

```
                                        .addComponent(bestE)
                                        .addComponent(goodE)
                                        .addComponent(okayE))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                        .addComponent(tornado)
                                        .addComponent(tBest)
                                        .addComponent(tGood)
                                        .addComponent(tOkay)
                                        )

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.TRAILING)
                                        .addComponent(bestT)
                                        .addComponent(goodT)
                                        .addComponent(okayT))
                                );
        layout.setVerticalGroup(
                layout.createSequentialGroup()
                .addComponent(Title)

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                    .addComponent(hurricain)
                                    .addComponent(earthquake)
                                    .addComponent(tornado))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                    .addComponent(hBest)
                                    .addComponent(bestH)
                                    .addComponent(eBest)
                                    .addComponent(bestE)
                                    .addComponent(tBest)
                                    .addComponent(bestT))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                    .addComponent(hGood)
                                    .addComponent(goodH)
                                    .addComponent(eGood)
                                    .addComponent(goodE)
                                    .addComponent(tGood)
                                    .addComponent(goodT))
```

```
    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(hOkay)
                                        .addComponent(okayH)
                                        .addComponent(eOkay)
                                        .addComponent(okayE)
                                        .addComponent(tOkay)
                                        .addComponent(okayT))

    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(Back))
                        );
                        pack();
                        setLocationRelativeTo(null);
                        setResizable(false);

    // Sets total strength of building
    }
    public void total(int total, int type) {
            endTot = total - 25;
            stotal = total;
            Damage h = new Damage();
h.setVisible(true);
this.dispose();
return;
    }
    private void BackActionPreformed(ActionEvent evt) {
            StartUI g = new StartUI();
            g.setVisible(true);
            this.dispose();
    }
    public void BestTorn(ActionEvent evt) {
            total(30, 0);

    }
    public void okayTorn(ActionEvent evt) {
            total(10, 0);
    }
    public void GoodTorn(ActionEvent evt) {
            total(20, 0);
```

```
	}
	public void Okayert(ActionEvent evt) {
		total(10, 1);
	}
	public void Goodert(ActionEvent evt) {
		total(20, 1);
	}
	public void Bestert(ActionEvent evt) {
		total(30, 1);
	}
	public void Okayher(ActionEvent evt) {
		total(10, 2);
	}
	public void Goodher(ActionEvent evt) {
		total(20, 2);
	}
	public void Besther(ActionEvent evt) {
		total(30, 2);
	}

}
```

Damage.java

```
package Disaster.Simulator;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class Damage extends JFrame{
	public Damage() {
		createmenu();
	}
	public void createmenu() {

		int totBe = BuildingModels.stotal;
		// Calculates the final score for each building
		int totAf = BuildingModels.endTot;
		// based on the original value each building is
```

```
        JLabel Title = new JLabel("Building Overview");
// given and the damage inflicted by each natural
        JLabel before = new JLabel("Total Before");
// disaster
        JLabel totB = new JLabel(""+totBe);
        JLabel totA = new JLabel(""+totAf);
        JLabel after = new JLabel("Total After");
        JLabel scale = new JLabel("Scale");
        JLabel good = new JLabel("Above 0: Good");
        JLabel okay = new JLabel("Below 0: Okay");
        JLabel ummm = new JLabel("Below -10: Bad");
        JButton restart = new JButton("Restart");

        Title.setFont(new Font("Tahoma", 68, 24));
        before.setFont(new Font("Tahoma", 78, 18));
        after.setFont(new Font("Tahoma", 78, 18));

        restart.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent evt) {
                        rest(evt);
                }
        });

        // Sets the layout for the damage page
        GroupLayout layout = new GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setAutoCreateGaps(true);
        layout.setAutoCreateContainerGaps(true);
        layout.setHorizontalGroup(
                layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                .addComponent(Title)

                                        .addComponent(before)
                                        .addComponent(totB))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.LEADING)
                                        .addComponent(scale)
                                        .addComponent(good)
```

```
                                        .addComponent(okay)
                                        .addComponent(ummm))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                        .addComponent(after)
                                        .addComponent(totA))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                        .addComponent(restart))
        );
        layout.setVerticalGroup(
                layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                    .addComponent(Title))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(before)
                                        .addComponent(after))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(totB)
                                        .addComponent(totA))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(scale))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(good))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(okay))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(ummm))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(restart))

        );
```

```
        pack();
        setLocationRelativeTo(null);
        setResizable(false);
           }
           public void rest(ActionEvent evt) {
                   StartUI s = new StartUI();
                   s.setVisible(true);
                   this.dispose();
           }
}
```

DamagePage.java

```
package Disaster.Simulator;

import javax.swing.*;
import java.awt.*;
import java.awt.event.*;

public class DamagePage extends JFrame{
        public DamagePage() {
                createmenu();
        }
        public void createmenu() {

                int totBe = NewBuilding.sTotal;
                        // Calculates the final score for each building
                int totAf = NewBuilding.endTot;
                        // based on the original value each building is
                JLabel Title = new JLabel("Building Overview");
        // given and the damage inflicted by each natural
                JLabel before = new JLabel("Total Before");
        // disaster
                JLabel totB = new JLabel(""+totBe);
                JLabel totA = new JLabel(""+totAf);
                JLabel after = new JLabel("Total After");
                JLabel scale = new JLabel("Scale");
                JLabel good = new JLabel("Above 0: Good");
                JLabel okay = new JLabel("Below 0: Okay");
                JLabel ummm = new JLabel("Below -10: Bad");
                JButton restart = new JButton("Restart");
```

```
        Title.setFont(new Font("Tahoma", 68, 24));
        before.setFont(new Font("Tahoma", 78, 18));
        after.setFont(new Font("Tahoma", 78, 18));

        restart.addActionListener(new ActionListener() {
                public void actionPerformed(ActionEvent evt) {
                        rest(evt);
                }
        });

        // Sets the layout for the damage page
        GroupLayout layout = new GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setAutoCreateGaps(true);
        layout.setAutoCreateContainerGaps(true);
        layout.setHorizontalGroup(
                layout.createSequentialGroup()

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                .addComponent(Title)

                                        .addComponent(before)
                                        .addComponent(totB))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.LEADING)
                                        .addComponent(scale)
                                        .addComponent(good)
                                        .addComponent(okay)
                                        .addComponent(ummm))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                        .addComponent(after)
                                        .addComponent(totA))

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                        .addComponent(restart))
        );
        layout.setVerticalGroup(
                layout.createSequentialGroup()
```

```
            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(Title))

            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                            .addComponent(before)
                                            .addComponent(after))

            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                            .addComponent(totB)
                                            .addComponent(totA))

            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                            .addComponent(scale))

            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                            .addComponent(good))

            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                            .addComponent(okay))

            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                            .addComponent(ummm))

            .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                            .addComponent(restart))

                    );
        pack();
        setLocationRelativeTo(null);
        setResizable(false);
           }
           public void rest(ActionEvent evt) {
                    StartUI s = new StartUI();
                    s.setVisible(true);
                    this.dispose();
           }
}
```

NewBuilding.java

```
package Disaster.Simulator;

import java.awt.*;
import java.awt.event.ActionEvent;
import java.awt.event.ActionListener;

import javax.swing.*;

public class NewBuilding extends JFrame {

	// Defines
	static int Type;
	static int sTotal;
	static int endTot;
	int FTT;
	int WTT;
	int TWT;
	int TRT;
	int WCT;
	int TDT;
	int RST;
	int TFT;
	int FTot;
	int HTot;
	int TST;
	int GDT;
	int size;

	// Populating arrays for different building aspects
	private String[] FoundationTypeL = { "",
			"Crawlspace",
			"Basement",
			"Slab",
			"Pile and Girder",
			"Crawlspace, Basement, or Slab with Retrofitting"
	};
	private String[] WindowTypeL = { "",
			"Impact Resistant Glass",
			"Double-Pane Glass",
			"Regular with Hurricane Film",
```

```
                "Regular with Storm Shudders",
                "Tempered Glass"
};
private String[] TypeWallL = { "",
                "Solid Concrete Walls",
                "Traditional Solid Masonry",
                "Wood",
                "Modern Solid Masonry",
                "Reinforced Wood",
                "Unreinforced Masonry (Traditional or Modern)",
                "Reinforced Masonry (Traditional or Modern)"
};
private String[] TypeRoofingL = { "",
                "Metal (steel, aluminum, tile and copper)",
                "Slate",
                "Clay & Concrete Tiles",
                "Wood shingles and shakes",
                "Asphalt shingles"
};
private String[] WallConectionL = { "",
                "Hurricane Clips and Straps",
                "Threaded Rods",
                "Cable-Tite Home Tie-Down Systems",
                "No Reinforcements",
                "Braced Wall Panels",
                "Continuous (Wood) Structural Panel Sheathing",
                "Wood Structural Panel Sheathed Walls with Hold-Down Connections"
};
private String[] TypeDoorL = { "",
                "Timber/Wood Door",
                "Battened and Ledged Door",
                "Glass Door",
                "Steel Door",
                "Fiberglass Door",
                "Aluminum Door"
};
private String[] RoofShapeL = { "",
                "Gable Roof",
                "Hip Roof",
                "Flat Roof"
```

```
};
private String[] TypeFlooringL = { "",
                "Ceramic Tile",
                "Porcelain Tile",
                "Hardwood",
                "Laminate",
                "Vinyl",
                "Marble",
                "Bamboo",
                "Carpet",
                "Stone"
};
private String[] FramingL = { "",
                "Timber",
                "Balloon",
                "Platform"
};
private String[] HeightL = { "",
                "One Story",
                "Two Stories",
                "Three-or-More Stories"
};
private String[] TypeSoilL = { "",
                "Soil Type A or Soil Type B",
                "Soil Type C",
                "Soil Type D",
                "Soil Type E"
};
private String[] GarageDoorL = { "","Wood",
                "Steel",
                "Aluminum",
                "No Garage Door",
                "Garage Door Braced with Plywood Panels and Steel Straps",
                "Any Material of Door with No Bracing"
};
private String[] DisasterL = { "",
                "Hurricane",
                "Tornado",
                "Earthquake"
};
```

```
int tSize = DisasterL.length;

public NewBuilding() {
	createMenu();
}


private void createMenu() {

	JLabel FT = new JLabel("Foundation Type:");
	//
	JLabel WT = new JLabel("Window Type:");
	JLabel TW = new JLabel("Type of Walls:");
	JLabel TR = new JLabel("Type of Roofing:");
	JLabel WC = new JLabel("Ceilings and Walls Conection:");
	JLabel TD = new JLabel("Type of Doors:");
	JLabel RS = new JLabel("Roof Shape");
	JLabel TF = new JLabel("Type of Flooring:");
	JLabel F = new JLabel("Framing:");
	JLabel H = new JLabel("Height:");
	JLabel TS = new JLabel("Type of Soil:");
	JLabel GD = new JLabel("Garage Door:");
	JLabel type = new JLabel("Type of Disaster:");
	JLabel Title = new JLabel("Design Your Building");
	JComboBox FoundationType = new JComboBox(FoundationTypeL);
	JComboBox WindowType = new JComboBox(WindowTypeL);
	JComboBox TypeWall = new JComboBox(TypeWallL);
	JComboBox TypeRoofing = new JComboBox(TypeRoofingL);
	JComboBox WallConection = new JComboBox(WallConectionL);
	JComboBox TypeDoor = new JComboBox(TypeDoorL);
	JComboBox RoofShape = new JComboBox(RoofShapeL);
	JComboBox TypeFlooring = new JComboBox(TypeFlooringL);
	JComboBox Framing = new JComboBox(FramingL);
	JComboBox Height = new JComboBox(HeightL);
	JComboBox TypeSoil = new JComboBox(TypeSoilL);
	JComboBox GarageDoor = new JComboBox(GarageDoorL);
	JComboBox Disaster = new JComboBox(DisasterL);
	Button Next = new Button("Next");
	Button Back = new Button("Back");
```

```
FoundationType.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                FTActionPreformed(evt);
        }

});
WindowType.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                WTActionPreformed(evt);
        }

});
TypeWall.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                TWActionPreformed(evt);
        }

});
TypeRoofing.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                TRActionPreformed(evt);
        }

});
WallConection.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                WCActionPreformed(evt);
        }

});
TypeDoor.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                TDActionPreformed(evt);
        }

});
RoofShape.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                RSActionPreformed(evt);
```

```
        }

});
TypeFlooring.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                TFActionPreformed(evt);
        }

});
Framing.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                FActionPreformed(evt);
        }

});
Height.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                HActionPreformed(evt);
        }

});
TypeSoil.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                TSActionPreformed(evt);
        }

});
GarageDoor.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                GDActionPreformed(evt);
        }

});
Disaster.addActionListener(new ActionListener() {
        public void actionPerformed(ActionEvent evt) {
                DActionPreformed(evt);
        }

});
```

```
                setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);

                Title.setFont(new java.awt.Font("Tahoma", 0, 24));
                Title.setHorizontalAlignment(javax.swing.SwingConstants.CENTER);
                Title.setVerticalAlignment(javax.swing.SwingConstants.BOTTOM);

                Next.setFont(new java.awt.Font("Tahoma", 0, 18));
                Next.setSize(new Dimension(30,20));
                Next.addActionListener(new java.awt.event.ActionListener() {
                        public void actionPerformed(java.awt.event.ActionEvent evt) {
                                DisastersActionPreformed(evt);
                        }
                });

                Back.setFont(new Font("Tahoma", 0, 18));
                Back.setSize(new Dimension(30,20));
                Back.addActionListener(new ActionListener() {
                        public void actionPerformed(ActionEvent evt) {
                                BackActionPreformed(evt);
                        }

                });

                // Sets layout for NewBuilding page
                javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
        getContentPane().setLayout(layout);
        layout.setAutoCreateGaps(true);
        layout.setAutoCreateContainerGaps(true);
        layout.linkSize(SwingConstants.HORIZONTAL, Back, Next);
        layout.setHorizontalGroup(
                layout.createSequentialGroup()

          .addGroup(layout.createParallelGroup(GroupLayout.Alignment.LEADING)
                                                .addComponent(Title)
                                                .addGap(100, 100, 100)
                                                .addComponent(type)
                                                .addComponent(FT)
```

```
                                        .addComponent(WT)
                                        .addComponent(TW)
                                        .addComponent(TR)
                                        .addComponent(WC)
                                        .addComponent(TD)
                                        .addComponent(RS)
                                        .addComponent(TF)
                                        .addComponent(F)
                                        .addComponent(H)
                                        .addComponent(TS)
                                        .addComponent(GD)
                                        .addGroup(layout.createSequentialGroup()

    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.TRAILING)
                                                    .addComponent(Back)
                                                    ))

                                        )

    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.CENTER)
                                        .addComponent(Disaster)
                                        .addComponent(FoundationType)
                                        .addComponent(WindowType)
                                        .addComponent(TypeWall)
                                        .addComponent(TypeRoofing)
                                        .addComponent(WallConection)
                                        .addComponent(TypeDoor)
                                        .addComponent(RoofShape)
                                        .addComponent(TypeFlooring)
                                        .addComponent(Framing)
                                        .addComponent(Height)
                                        .addComponent(TypeSoil)
                                        .addComponent(GarageDoor)
                                        .addComponent(Next)
                                        )

            );

// Sets the layout for the NewBuilding page
layout.setVerticalGroup(
```

```
        layout.createSequentialGroup()

                .addComponent(Title)
                .addGap(20, 20, 20)

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                .addComponent(type)
                                .addComponent(Disaster)
                                )

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                .addComponent(FT)
                                .addComponent(FoundationType)
                                )

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                .addComponent(WT)
                                .addComponent(WindowType)
                                )

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                .addComponent(TW)
                                .addComponent(TypeWall)
                                )

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                .addComponent(TR)
                                .addComponent(TypeRoofing)
                                )

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                .addComponent(WC)
                                .addComponent(WallConection)
                                )

.addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                .addComponent(TD)
                                .addComponent(TypeDoor)
                                )
```

```
    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(RS)
                                        .addComponent(RoofShape)
                                        )

    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(TF)
                                        .addComponent(TypeFlooring)
                                        )

    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(F)
                                        .addComponent(Framing)
                                        )

    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(H)
                                        .addComponent(Height)
                                        )

    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(TS)
                                        .addComponent(TypeSoil)
                                        )

    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(GD)
                                        .addComponent(GarageDoor)
                                        )

    .addGroup(layout.createParallelGroup(GroupLayout.Alignment.BASELINE)
                                        .addComponent(Back)
                                        .addComponent(Next)
                                        )
            );


pack();
setLocationRelativeTo(null);
```

```
setResizable(false);
    }

    // Assigns strength to foundation types
    public void FTActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String FTn = (String)cb.getSelectedItem();

if(Type == 0) {
    if(FTn.equals("Crawlspace")) {
            FTT = 1;
    }
    else if(FTn.equals("Basement")) {
            FTT = 2;
    }
    else if(FTn.equals("Slab")) {
            FTT = 1;
    }
    else if(FTn.equals("Pile and Girder")) {
            FTT = 3;
    }
    else if(FTn.equals("Crawlspace, Basement, or Slab with Retrofitting")) {
            FTT = 3;
    }
}
else if (Type == 1) {
    if(FTn.equals("Crawlspace")) {
            FTT = 3;
    }
    else if(FTn.equals("Basement")) {
            FTT = 3;
    }
    else if(FTn.equals("Slab")) {
            FTT = 2;
    }
    else if(FTn.equals("Pile and Girder")) {
            FTT = 1;
    }
    else if(FTn.equals("Crawlspace, Basement, or Slab with Retrofitting")) {
            FTT = 3;
```

```
    }
}
else if (Type == 2) {
    if(FTn.equals("Crawlspace")) {
            FTT = 2;
    }
    else if(FTn.equals("Basement")) {
            FTT = 2;
    }
    else if(FTn.equals("Slab")) {
            FTT = 2;
    }
    else if(FTn.equals("Pile and Girder")) {
            FTT = 1;
    }
    else if(FTn.equals("Crawlspace, Basement, or Slab with Retrofitting")) {
            FTT = 3;
    }
}

    }

    // Adds strength to window type
    public void WTActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String WTn = (String)cb.getSelectedItem();

    if(Type == 0) {
            if(WTn.equals("Impact Resistant Glass")) {
                    WTT = 3;
            }
            else if(WTn.equals("Double-Pane Glass")) {
                    WTT = 1;
            }
            else if(WTn.equals("Regular with Hurricane Film")) {
                    WTT = 2;
            }
            else if(WTn.equals("Regular with Storm Shudders")) {
                    WTT = 1;
            }
```

```
        else if(WTn.equals("Tempered Glass")) {
                WTT = 3;
        }
}
else if (Type == 1) {
        if(WTn.equals("Impact Resistant Glass")) {
                WTT = 3;
        }
        else if(WTn.equals("Double-Pane Glass")) {
                WTT = 1;
        }
        else if(WTn.equals("Regular with Hurricane Film")) {
                WTT = 2;
        }
        else if(WTn.equals("Regular with Storm Shudders")) {
                WTT = 2;
        }
        else if(WTn.equals("Tempered Glass")) {
                WTT = 3;
        }
}
else if (Type == 2) {
        if(WTn.equals("Impact Resistant Glass")) {
                WTT = 2;
        }
        else if(WTn.equals("Double-Pane Glass")) {
                WTT = 1;
        }
        else if(WTn.equals("Regular with Hurricane Film")) {
                WTT = 3;
        }
        else if(WTn.equals("Regular with Storm Shudders")) {
                WTT = 3;
        }
        else if(WTn.equals("Tempered Glass")) {
                WTT = 3;
        }
}

}
```

```
   // Adds strength to types of walls
   public void TWActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String TWn = (String)cb.getSelectedItem();

if(Type == 0) {
   if(TWn.equals("Solid Concrete Walls ")) {
            TWT = 3;
   }
   else if(TWn.equals("Traditional Solid Masonry")) {
            TWT = 2;
   }
   else if(TWn.equals("Wood")) {
            TWT = 1;
   }
   else if(TWn.equals("Modern Solid Masonry")) {
            TWT = 3;
   }
   else if(TWn.equals("Reinforced Wood")) {
            TWT = 3;
   }
   else if(TWn.equals("Unreinforced Masonry (Traditional or Modern")) {
            TWT = 3;
   }
   else if(TWn.equals("Reinforced Masonry (Traditional or Modern")) {
            TWT = 3;
   }
}
else if (Type == 1) {
   if(TWn.equals("Solid Concrete Walls ")) {
            TWT = 3;
   }
   else if(TWn.equals("Traditional Solid Masonry")) {
            TWT = 3;
   }
   else if(TWn.equals("Wood")) {
            TWT = 1;
   }
   else if(TWn.equals("Modern Solid Masonry")) {
```

```
            TWT = 2;
    }
    else if(TWn.equals("Reinforced Wood")) {
            TWT = 3;
    }
    else if(TWn.equals("Unreinforced Masonry (Traditional or Modern")) {
            TWT = 3;
    }
    else if(TWn.equals("Reinforced Masonry (Traditional or Modern")) {
            TWT = 3;
    }
}
else if (Type == 2) {
    if(TWn.equals("Solid Concrete Walls ")) {
            TWT = 3;
    }
    else if(TWn.equals("Traditional Solid Masonry")) {
            TWT = 3;
    }
    else if(TWn.equals("Wood")) {
            TWT = 3;
    }
    else if(TWn.equals("Modern Solid Masonry")) {
            TWT = 3;
    }
    else if(TWn.equals("Reinforced Wood")) {
            TWT = 3;
    }
    else if(TWn.equals("Unreinforced Masonry (Traditional or Modern")) {
            TWT = 1;
    }
    else if(TWn.equals("Reinforced Masonry (Traditional or Modern")) {
            TWT = 2;
    }
}



    }

    // Adds strength to type of roofing
```

```
   public void TRActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String TRn = (String)cb.getSelectedItem();

if(Type == 0) {
   if(TRn.equals("Metal (steel, aluminum, tile and copper)")) {
            TRT = 3;
   }
   else if(TRn.equals("Slate")) {
            TRT = 3;
   }
   else if(TRn.equals("Wood")) {
            TRT = 3;
   }
   else if(TRn.equals("Clay & Concrete Tiles")) {
            TRT = 3;
   }
   else if(TRn.equals("Wood shingles and shakes")) {
            TRT = 1;
   }
   else if(TRn.equals("Asphalt shingles")) {
            TRT = 2;
   }
}
else if (Type == 1) {
   if(TRn.equals("Metal (steel, aluminum, tile and copper)")) {
            TRT = 3;
   }
   else if(TRn.equals("Slate")) {
            TRT = 3;
   }
   else if(TRn.equals("Wood")) {
            TRT = 3;
   }
   else if(TRn.equals("Clay & Concrete Tiles")) {
            TRT = 3;
   }
   else if(TRn.equals("Wood shingles and shakes")) {
            TRT = 1;
   }
```

```
    else if(TRn.equals("Asphalt shingles")) {
            TRT = 2;
    }
}
else if (Type == 2) {
    if(TRn.equals("Metal (steel, aluminum, tile and copper)")) {
            TRT = 3;
    }
    else if(TRn.equals("Slate")) {
            TRT = 1;
    }
    else if(TRn.equals("Wood")) {
            TRT = 3;
    }
    else if(TRn.equals("Clay & Concrete Tiles")) {
            TRT = 1;
    }
    else if(TRn.equals("Wood shingles and shakes")) {
            TRT = 2;
    }
    else if(TRn.equals("Asphalt shingles")) {
            TRT = 2;
    }
}



    }

    // Adds strength to ceiling and wall connections
    public void WCActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String WCn = (String)cb.getSelectedItem();
if (Type == 0) {
    if(WCn.equals("Hurricane Clips and Straps")) {
            WCT = 1;
    }
    else if(WCn.equals("Threaded Rods")) {
            WCT = 2;
    }
    else if(WCn.equals("Cable-Tite Home Tie-Down Systems")) {
```

```
                WCT = 3;
        }
        else if(WCn.equals("No Reinforcements")) {
                WCT = 1;
        }
        else if(WCn.equals("Braced Wall Panels")) {
                WCT = 3;
        }
        else if(WCn.equals("Continuous (Wood) Structural Panel Sheathing")) {
                WCT = 3;
        }
        else if(WCn.equals("Wood Structural Panel Sheathed Walls with Hold-Down
Connections")) {
                WCT = 3;
        }
      }
      else if (Type == 1) {
        if(WCn.equals("Hurricane Clips and Straps")) {
                WCT = 3;
        }
        else if(WCn.equals("Threaded Rods")) {
                WCT = 2;
        }
        else if(WCn.equals("Cable-Tite Home Tie-Down Systems")) {
                WCT = 3;
        }
        else if(WCn.equals("No Reinforcements")) {
                WCT = 1;
        }
        else if(WCn.equals("Braced Wall Panels")) {
                WCT = 3;
        }
        else if(WCn.equals("Continuous (Wood) Structural Panel Sheathing")) {
                WCT = 3;
        }
        else if(WCn.equals("Wood Structural Panel Sheathed Walls with Hold-Down
Connections")) {
                WCT = 3;
        }
      }
```

```
        else if (Type == 2) {
            if(WCn.equals("Hurricane Clips and Straps")) {
                    WCT = 3;
            }
            else if(WCn.equals("Threaded Rods")) {
                    WCT = 3;
            }
            else if(WCn.equals("Cable-Tite Home Tie-Down Systems")) {
                    WCT = 3;
            }
            else if(WCn.equals("No Reinforcements")) {
                    WCT = 1;
            }
            else if(WCn.equals("Braced Wall Panels")) {
                    WCT = 2;
            }
            else if(WCn.equals("Continuous (Wood) Structural Panel Sheathing")) {
                    WCT = 2;
            }
            else if(WCn.equals("Wood Structural Panel Sheathed Walls with Hold-Down
Connections")) {
                    WCT = 3;
            }
        }



            }

            // Adds strength to types of doors
            public void TDActionPreformed(ActionEvent evt) {
                    JComboBox cb = (JComboBox)evt.getSource();
        String TDn = (String)cb.getSelectedItem();
        if (Type == 0) {
            if(TDn.equals("Timber/Wood Door")) {
                    TDT = 3;
            }
            else if(TDn.equals("Battened and Ledged Door")) {
                    TDT = 3;
            }
            else if(TDn.equals("Glass Door")) {
```

```
            TDT = 1;
    }
    else if(TDn.equals("Steel Door")) {
            TDT = 2;
    }
    else if(TDn.equals("Fiberglass Door")) {
            TDT = 2;
    }
    else if(TDn.equals("Aluminum Door")) {
            TDT = 1;
    }

}
else if (Type == 1) {
    if(TDn.equals("Timber/Wood Door")) {
            TDT = 3;
    }
    else if(TDn.equals("Battened and Ledged Door")) {
            TDT = 3;
    }
    else if(TDn.equals("Glass Door")) {
            TDT = 1;
    }
    else if(TDn.equals("Steel Door")) {
            TDT = 2;
    }
    else if(TDn.equals("Fiberglass Door")) {
            TDT = 2;
    }
    else if(TDn.equals("Aluminum Door")) {
            TDT = 1;
    }

}
else if (Type == 2) {
    if(TDn.equals("Timber/Wood Door")) {
            TDT = 2;
    }
    else if(TDn.equals("Battened and Ledged Door")) {
            TDT = 3;
```

```
    }
    else if(TDn.equals("Glass Door")) {
            TDT = 1;
    }
    else if(TDn.equals("Steel Door")) {
            TDT = 3;
    }
    else if(TDn.equals("Fiberglass Door")) {
            TDT = 3;
    }
    else if(TDn.equals("Aluminum Door")) {
            TDT = 2;
    }

}

    }

    // Adds strength to roof shapes
    public void RSActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String RSn = (String)cb.getSelectedItem();
if (Type == 0) {
    if(RSn.equals("Gable Roof")) {
            RST = 2;
    }
    else if(RSn.equals("Hip Roof")) {
            RST = 3;
    }
    else if(RSn.equals("Flat Roof")) {
            RST = 1;
    }

}
else if (Type == 1) {
    if(RSn.equals("Gable Roof")) {
            RST = 2;
    }
    else if(RSn.equals("Hip Roof")) {
            RST = 3;
```

```
    }
    else if(RSn.equals("Flat Roof")) {
            RST = 1;
    }

}
else if (Type == 2) {
    if(RSn.equals("Gable Roof")) {
            RST = 3;
    }
    else if(RSn.equals("Hip Roof")) {
            RST = 3;
    }
    else if(RSn.equals("Flat Roof")) {
            RST = 3;
    }

}

    }

    // Adds strength to type of flooring
    public void TFActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String TFn = (String)cb.getSelectedItem();
if (Type == 0) {
    if(TFn.equals("Ceramic Tile")) {
            TFT = 3;
    }
    else if(TFn.equals("Porcelain Tile")) {
            TFT = 3;
    }
    else if(TFn.equals("Hardwood")) {
            TFT = 1;
    }
    else if(TFn.equals("Laminate")) {
            TFT = 1;
    }
    else if(TFn.equals("Vinyl")) {
            TFT = 2;
```

```
    }
    else if(TFn.equals("Marble")) {
            TFT = 3;
    }
    else if(TFn.equals("Bamboo")) {
            TFT = 1;
    }
    else if(TFn.equals("Carpet")) {
            TFT = 1;
    }
    else if(TFn.equals("Stone")) {
            TFT = 2;
    }
}
else if (Type == 1) {
    if(TFn.equals("Ceramic Tile")) {
            TFT = 3;
    }
    else if(TFn.equals("Porcelain Tile")) {
            TFT = 3;
    }
    else if(TFn.equals("Hardwood")) {
            TFT = 3;
    }
    else if(TFn.equals("Laminate")) {
            TFT = 3;
    }
    else if(TFn.equals("Vinyl")) {
            TFT = 3;
    }
    else if(TFn.equals("Marble")) {
            TFT = 3;
    }
    else if(TFn.equals("Bamboo")) {
            TFT = 3;
    }
    else if(TFn.equals("Carpet")) {
            TFT = 3;
    }
    else if(TFn.equals("Stone")) {
```

```
            TFT = 3;
    }
}
else if (Type == 2) {
    if(TFn.equals("Ceramic Tile")) {
            TFT = 3;
    }
    else if(TFn.equals("Porcelain Tile")) {
            TFT = 3;
    }
    else if(TFn.equals("Hardwood")) {
            TFT = 3;
    }
    else if(TFn.equals("Laminate")) {
            TFT = 3;
    }
    else if(TFn.equals("Vinyl")) {
            TFT = 3;
    }
    else if(TFn.equals("Marble")) {
            TFT = 3;
    }
    else if(TFn.equals("Bamboo")) {
            TFT = 3;
    }
    else if(TFn.equals("Carpet")) {
            TFT = 3;
    }
    else if(TFn.equals("Stone")) {
            TFT = 3;
    }
}

    }

    // Adds strength to framing
    public void FActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String Fn = (String)cb.getSelectedItem();
if (Type == 0) {
```

```
    if(Fn.equals("Timber")) {
            FTot = 3;
    }
    else if(Fn.equals("Balloon")) {
            FTot = 3;
    }
    else if(Fn.equals("Platform")) {
            FTot = 3;
    }

}
else if (Type == 1) {
    if(Fn.equals("Timber")) {
            FTot = 3;
    }
    else if(Fn.equals("Balloon")) {
            FTot = 1;
    }
    else if(Fn.equals("Platform")) {
            FTot = 2;
    }

}
else if (Type == 2) {
    if(Fn.equals("Timber")) {
            FTot = 3;
    }
    else if(Fn.equals("Balloon")) {
            FTot = 2;
    }
    else if(Fn.equals("Platform")) {
            FTot = 1;
    }

}
    }

    // Adds strength to house height
    public void HActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
```

```
String HAn = (String)cb.getSelectedItem();
if (Type == 0) {
   if(HAn.equals("One Story")) {
            HTot = 3;
   }
   else if(HAn.equals("Two Story")) {
            HTot = 3;
   }
   else if(HAn.equals("Three-or-More Stories")) {
            HTot = 3;
   }

}
else if (Type == 1) {
   if(HAn.equals("One Story")) {
            HTot = 3;
   }
   else if(HAn.equals("Two Story")) {
            HTot = 2;
   }
   else if(HAn.equals("Three-or-More Stories")) {
            HTot = 1;
   }

}
else if (Type == 2) {
   if(HAn.equals("One Story")) {
            HTot = 3;
   }
   else if(HAn.equals("Two Story")) {
            HTot = 2;
   }
   else if(HAn.equals("Three-or-More Stories")) {
            HTot = 1;
   }

}
   }

   // Adds strength to soils type
```

```
    public void TSActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String TSn = (String)cb.getSelectedItem();
if (Type == 0) {
    if(TSn.equals("Soil Type A or Soil Type B")) {
            TST = 3;
    }
    else if(TSn.equals("Soil Type C")) {
            TST = 3;
    }
    else if(TSn.equals("Soil Type D")) {
            TST = 3;
    }
    else if(TSn.equals("Soil Type E")) {
            TST = 3;
    }

}
else if (Type == 1) {
    if(TSn.equals("Soil Type A or Soil Type B")) {
            TST = 3;
    }
    else if(TSn.equals("Soil Type C")) {
            TST = 3;
    }
    else if(TSn.equals("Soil Type D")) {
            TST = 3;
    }
    else if(TSn.equals("Soil Type E")) {
            TST = 3;
    }

}
else if (Type == 2) {
    if(TSn.equals("Soil Type A or Soil Type B")) {
            TST = 3;
    }
    else if(TSn.equals("Soil Type C")) {
            TST = 2;
    }
```

```
    else if(TSn.equals("Soil Type D")) {
            TST = 1;
    }
    else if(TSn.equals("Soil Type E")) {
            TST = 1;
    }

}
    }

    // Adds strength to garage door
    public void GDActionPreformed(ActionEvent evt) {
            JComboBox cb = (JComboBox)evt.getSource();
String GDn = (String)cb.getSelectedItem();
if (Type == 0) {
    if(GDn.equals("Wood")) {
            GDT = 1;
    }
    else if(GDn.equals("Steel")) {
            GDT = 2;
    }
    else if(GDn.equals("Aluminum")) {
            GDT = 3;
    }
    else if(GDn.equals("No Garage Door")) {
            GDT = 3;
    }
    else if(GDn.equals("Garage Door Braced with Plywood Panels and Steel Straps")) {
            GDT = 3;
    }
    else if(GDn.equals("Any Material of Door with No Bracing")) {
            GDT = 3;
    }

}
else if (Type == 1) {
    if(GDn.equals("Wood")) {
            GDT = 1;
    }
    else if(GDn.equals("Steel")) {
```

```
            GDT = 2;
    }
    else if(GDn.equals("Aluminum")) {
            GDT = 3;
    }
    else if(GDn.equals("No Garage Door")) {
            GDT = 3;
    }
    else if(GDn.equals("Garage Door Braced with Plywood Panels and Steel Straps")) {
            GDT = 3;
    }
    else if(GDn.equals("Any Material of Door with No Bracing")) {
            GDT = 3;
    }

}
else if (Type == 2) {
    if(GDn.equals("Wood")) {
            GDT = 3;
    }
    else if(GDn.equals("Steel")) {
            GDT = 3;
    }
    else if(GDn.equals("Aluminum")) {
            GDT = 3;
    }
    else if(GDn.equals("No Garage Door")) {
            GDT = 3;
    }
    else if(GDn.equals("Garage Door Braced with Plywood Panels and Steel Straps")) {
            GDT = 2;
    }
    else if(GDn.equals("Any Material of Door with No Bracing")) {
            GDT = 1;
    }

}

    }
```

```
        // Adds damage to each natural disaster. This
        // damage is applied after each house score
        // is calculated to give you a final score.
        public void DActionPreformed(ActionEvent evt) {
                JComboBox cb = (JComboBox)evt.getSource();
    String Dn = (String)cb.getSelectedItem();

    if(Dn.equals ("Hurricane") ){
        Type = 0;
    }
    else if(Dn.equals("Tornado")) {
        Type = 1;
    }
    else if(Dn.equals("Earthquake")) {
        Type = 2;
    }

        }

        // Adds up the total score for each house
        // and assigns a final score determining
        // how well your house would hold up against
        // the natural disaster that you chose.
        private void DisastersActionPreformed(java.awt.event.ActionEvent evt) {
    sTotal = FTT + WTT + TWT + TRT + WCT + TDT + RST + TFT + FTot + HTot + TST +
GDT;
    endTot = sTotal - 25;
    DamagePage h = new DamagePage();
    h.setVisible(true);
    this.dispose();
  }

        // Creates a back button that allows you
        // to return to the last page.
        private void BackActionPreformed(ActionEvent evt) {
                StartUI g = new StartUI();
                g.setVisible(true);
                this.dispose();
        }
```

```
}
```

StartUI.java

```
package Disaster.Simulator;

import java.awt.*;
import javax.swing.*;


public class StartUI extends javax.swing.JFrame {



        public StartUI() {
                createMenu();
        }

        // Creates the menu page.
        private void createMenu() {
                JLabel Title = new JLabel();

                Button NewBuilding = new Button();
                Button BuildingModels = new Button();
                Button Credits = new Button();


                setDefaultCloseOperation(javax.swing.WindowConstants.EXIT_ON_CLOSE);
                setPreferredSize(new java.awt.Dimension(554, 458));
                // Sets dimensions of menu, close
                setLocationRelativeTo(null);
                        //operations, and centers in screen

                Title.setFont(new java.awt.Font("Tahoma", 1, 20));
                Title.setText("Disaster Simulator");

                NewBuilding.setFont(new java.awt.Font("Tahoma", 1, 16));
        // Sets the font and creates a
                NewBuilding.setLabel("NewBuilding");
                        // button on the home screen
                NewBuilding.addActionListener(new java.awt.event.ActionListener() {
```

```
                public void actionPerformed(java.awt.event.ActionEvent evt) {
                    NewBuildingActionPreformed(evt);
                }
            });

            BuildingModels.setFont(new java.awt.Font("Tahoma", 1, 16));
        // Sets the font and creates a
            BuildingModels.setLabel("BuildingModels");
                // button on the home screen
            BuildingModels.addActionListener(new java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt) {
                    BuildingModelsActionPreformed(evt);
                }
            });

            Credits.setFont(new java.awt.Font("Tahoma", 1, 16));
        // Set the font and creates a
            Credits.setLabel("Credits");
                // button on the home screen
            Credits.addActionListener(new java.awt.event.ActionListener() {
                public void actionPerformed(java.awt.event.ActionEvent evt) {
                    //BuildingModelsActionPreformed(evt);
                }
            });

            // Sets the layout of buttons and labels to the jframe
            javax.swing.GroupLayout layout = new
javax.swing.GroupLayout(getContentPane());
      getContentPane().setLayout(layout);
      layout.setHorizontalGroup(
         layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
         .addGroup(layout.createSequentialGroup()
            .addGap(173, 173, 173)

.addGroup(layout.createParallelGroup(javax.swing.GroupLayout.Alignment.CENTER)
               .addComponent(NewBuilding, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
        // Adds NewBuilding to the layout
```

```
                .addComponent(BuildingModels, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE) //
Adds BuildingModels to the layout
                .addComponent(Credits, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                                                                        // Adds Credits to the
layout
                .addComponent(Title))

            .addContainerGap(195, Short.MAX_VALUE))
        );

                layout.setVerticalGroup(
                layout.createParallelGroup(javax.swing.GroupLayout.Alignment.LEADING)
                .addGroup(layout.createSequentialGroup()
                    .addGap(13, 13, 13)
                    .addComponent(Title, javax.swing.GroupLayout.PREFERRED_SIZE, 78,
javax.swing.GroupLayout.PREFERRED_SIZE)
                                        // Adds Title to the layout
                    .addGap(9, 9, 9)
                    .addComponent(NewBuilding, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
            // Adds NewBuilding to the layout
                    .addGap(32, 32, 32)
                    .addComponent(BuildingModels,
javax.swing.GroupLayout.PREFERRED_SIZE, javax.swing.GroupLayout.DEFAULT_SIZE,
javax.swing.GroupLayout.PREFERRED_SIZE)             // Adds BuildingModels to the layout
                    .addGap(38, 38, 38)
                    .addComponent(Credits, javax.swing.GroupLayout.PREFERRED_SIZE,
javax.swing.GroupLayout.DEFAULT_SIZE, javax.swing.GroupLayout.PREFERRED_SIZE)
                // Adds Credits to the layout
                    .addContainerGap(107, Short.MAX_VALUE))
            );

                pack();
                setLocationRelativeTo(null);
                setResizable(false);
        }

        private void NewBuildingActionPreformed(java.awt.event.ActionEvent evt) {
```

```
        NewBuilding s = new NewBuilding();
                                    // Changes page when the NewBuilding
        s.setVisible(true);
                                                    // button is clicked
        this.dispose();

    }

        private void BuildingModelsActionPreformed(java.awt.event.ActionEvent evt) {

                BuildingModels a = new BuildingModels();
                                    // Changes page when the BuildingModels
                a.setVisible(true);
                                                            // button is clicked
                this.dispose();
        }

        public static void main(String[] args) {
                                    // Runs the entire program
                java.awt.EventQueue.invokeLater(new Runnable() {
                        public void run() {
                                new StartUI().setVisible(true);
                        }
                });

        }

}
```