

Algorithms for Calculating a Building's Waste

Drew Olson

This paper examines the amount of waste or unutilized material a particular design will produce. This is being analyzed by the author through the investigation and use of multiple software systems. The ideal intent is the creation of such a program that will be able to inform the designer of the waste his or her particular design is going to be producing with a real time report. This product could then be implemented into BIM programs such as Revit by Autodesk or any other DXF file formatted software.

Introduction:

The philosophical premise behind this investigation is to get an understanding of how much waste is being produced from a specific design. Knowledge of the waste a design is going to produce will help bring an understanding of the designer's impacts on the environment. Designers need to utilize the planet's resources in the most efficient way possible through the conservation of materials and the economic efficiency of resources. The professions' ideal intent is to build beautiful structures that can coexist in their environment and do not add to the building mounds of unutilized material in our landfills. As a student of architecture a conscious awareness of everyone's impacts on the environment should be fully understood for anyone to be a steward of the built environment which architects are creating.

This paper's goal is avoiding waste, but how is waste avoided? The resources that the building industry is using to produce the building materials are just being transformed from their original physical state. Viewing the world as a whole, materials are not technically producing waste unless the product can become hazardous to our health and environment, but it is the unutilized materials that are filling the landfills. How can one maximize their efficiency of building materials to minimize waste?

The current waste being produced from a construction site is based off a general percentage that is more or less just a rule of thumb than a true measure of the waste actually being produced. If the true percentage of waste could be determined by a BIM program, then the next step is to determine how this is produced. Can the total material used be calculated from an algorithm that could maximize the material used to produce minimal waste? The ramifications of such an algorithm would in turn reduce the quantity of materials in a building, allowing more money to be put toward a

higher quality of materials. The end result of this study would be the creation of a program that would calculate the most efficient procedure and layout of material based off a particular design.

Methodology

The formulation of an algorithm that will calculate the total waste being produced from a design in order to inform the designer; also given different circumstances, which layout of material will produce the most efficient end product. The process is going to begin with understanding all the variables and constants that are going to be within a specific component. With that information gathered it is creating a step by step process of going through the different constraints of a design to maximize the yield of material and formulate a calculated sum of unutilized material.

The goal is to create an algorithm that will calculate the total wasted materials for an individual building component; i.e. a floor system, wall system, and roof system. With the use of a Word document and recording the step by step process of constructing an algorithm for the computation of wasted material of an individual building component can be analyzed for a particular design through the use of pseudocode. The initial intent was to bring the algorithms that had been produced to a computer software programmer who could then implement the algorithms into a program such as Revit. As the research process unfolded, taking the step of working out and writing a program personally became more captivating. The program would allow an individual to analyze their design by the amount of wasted material that a particular design would be producing. The user would be able to make changes to their design, and, with the click of a button, the program could recalculate the wasted material of the altered design at real-time speed.

Process and Procedure

The process of this project began with researching what was currently out in the market today for such an application. A program that provided the most insight was Floor Estimator Pro. This program takes the floor plan and provides an accurate and quick estimation. The key concepts that were learned in reaching this program and a couple others like it were the constraints and variables that are associated with this component of the building system. From research of these programs, an exercise was done on graph paper. This was a great introduction to application the constraints and variables that were going to be in play moving forth within the research. Following these exercises background knowledge of the content information that would form the foundation of the research investigation was established. The resources and software programs used in this investigation were; Modeling Objects and Environments, Fundamental Algorithms: The Art of Computer Programming, Visual Basic Graphics Programming, electronic class posting by my professor Dr. Ganapathy Mahalingam, online lectures from MIT structured toward computer programming, Pseudocode Standard, Microsoft Word, Visio, and Visual Studio, Adobe's InDesign CS3, and Adobe's Photoshop CS3.

The process worked in a linear fashion: beginning with the hand drawn experiments on graph paper and then constructing the algorithm structured by the pseudocode standard. From the pseudocode the algorithm was refined and developed into a flow chart which introduced more variables and processes in a more detailed order. Once a flow chart was constructed, the decision of repeating this process for another building component or carrying this component into a computer language was addressed. Development and understanding of this algorithm were brought into Visual Basic to construct a program that could operate the simple function of outputting a square footage of

material declared as waste and producing a graphical representation of the floor plan (polygon) inputted into the program.

Results

Figure 1

Figures 1-4 are the four exercises that were developed the first week of research. They compared the origin point and direction of the material to find how the wasted material is created.

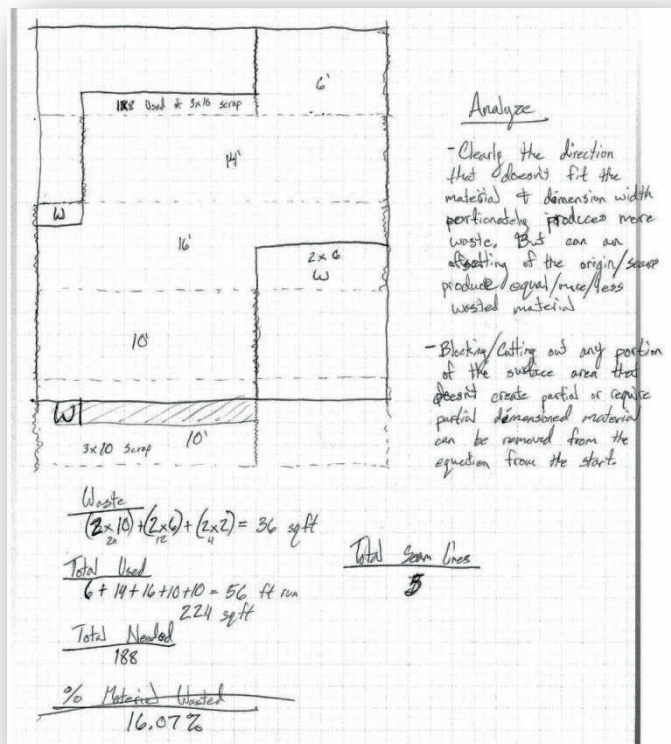
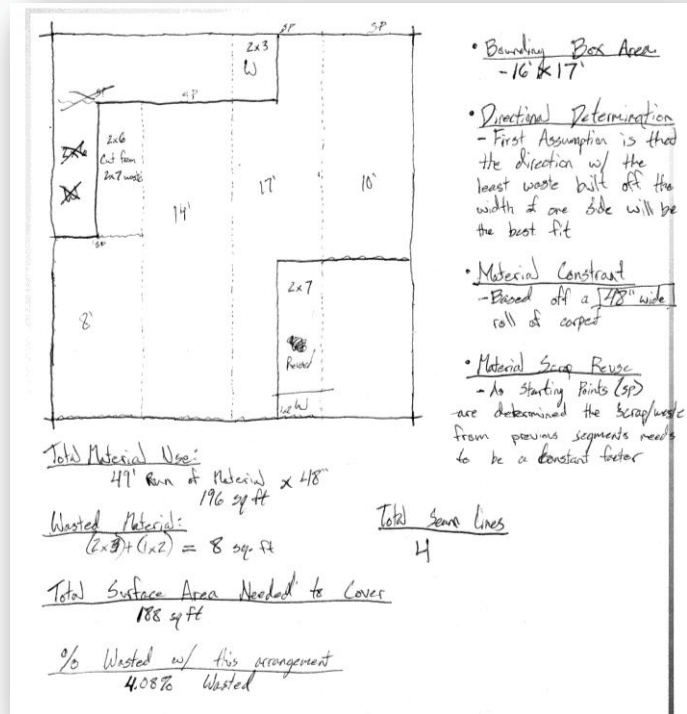


Figure 2

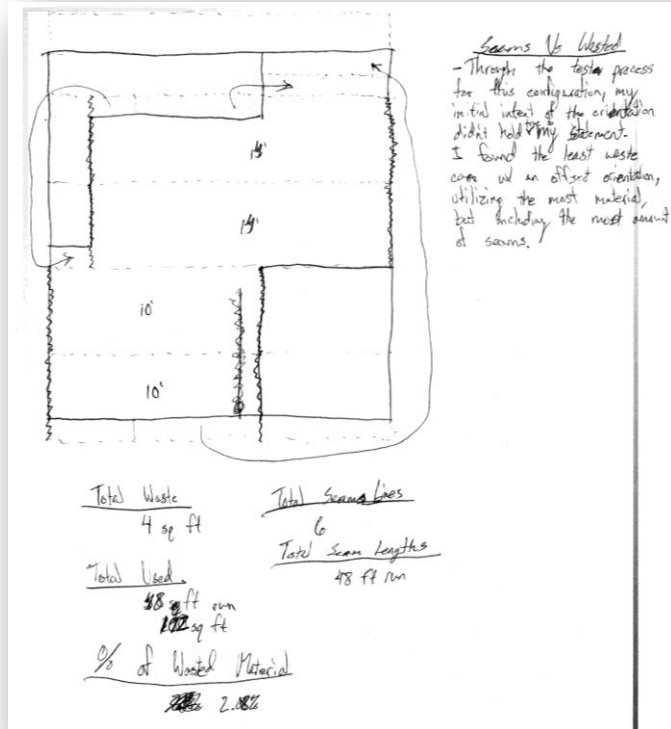
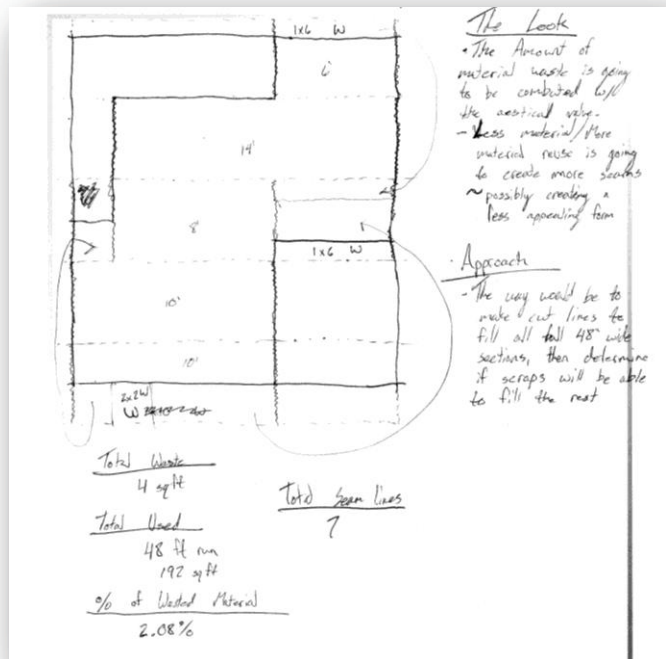


Figure 3

Figure 4



This process really engages one's thinking and became inherently important in the understanding of the step by step procedure required in creating an algorithm that is still being discovered at this stage in the research. Producing the possible outcomes and recording the findings were a relatively simple task at the beginning, but the knowledge of the pseudocode and computer programming languages became the mountain to climb ahead.

The following is a pseudocode written aimed towards the efficiency of carpeting a floor surface given any floor layout.

Start:

Origin = (0, 0)

OptimalMatUsage = $\frac{\text{Tot. Sq. Ft. of Floor Plan}}{12' \text{ (width of defined carpet)}}$

Efficiency Check = $\left\{ \frac{((\text{Tot. Ft. Run of Mat used}) - \text{OptimalMatUsage})}{\text{OptimalMatUsage}} \right\} * 100$

Defined Material = 12' x R (roll length)

Aesthetic Check = Sum Ft Run of segments

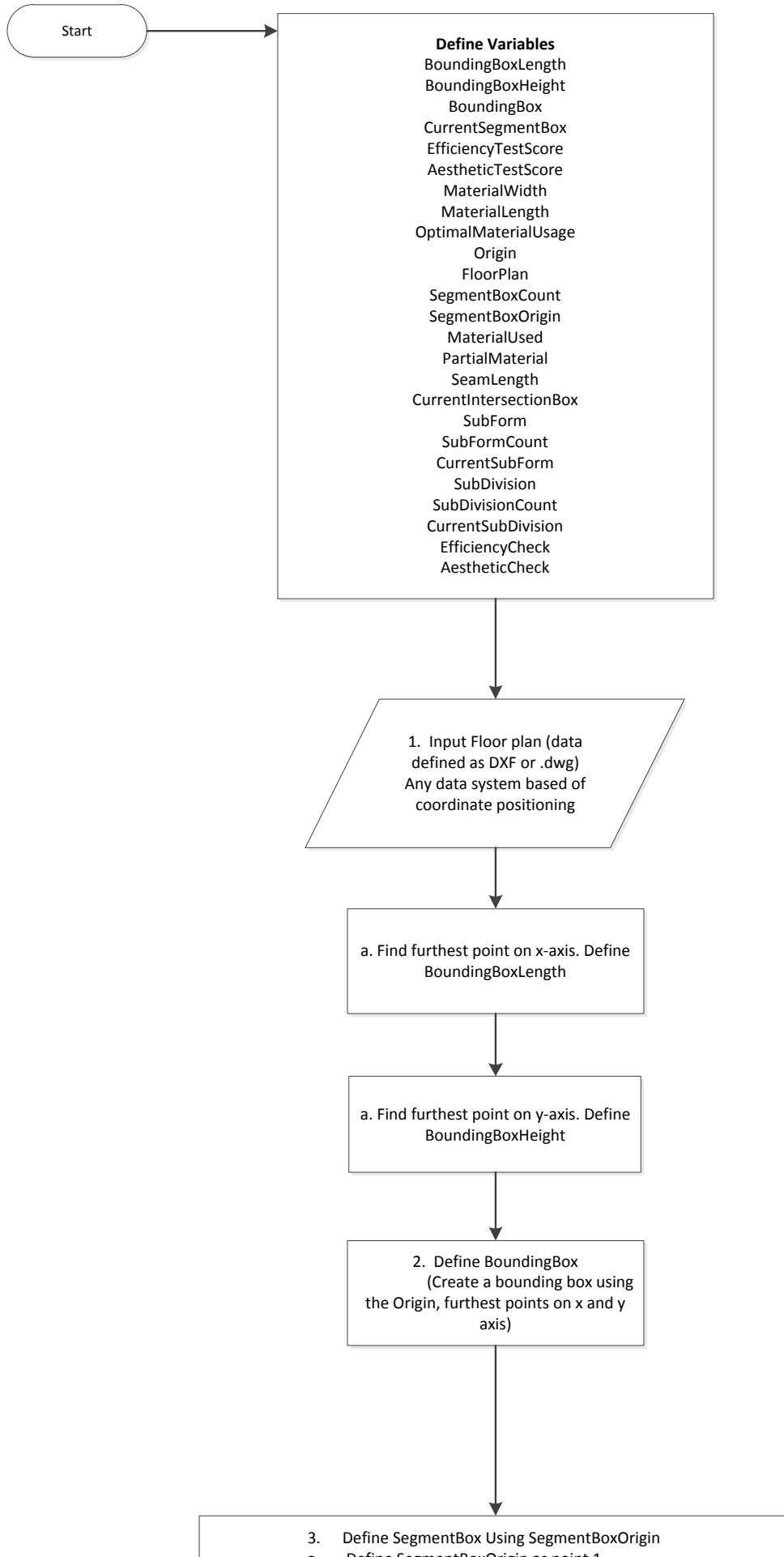
1. Input floor plan (data defined by as DXF or .dwg) or any data system based of coordinate positioning
 - a. Find furthest point on x-axis
 - i. Store in memory
 - b. Find furthest point on y-axis
 - i. Store in memory
2. Define Bounding Box

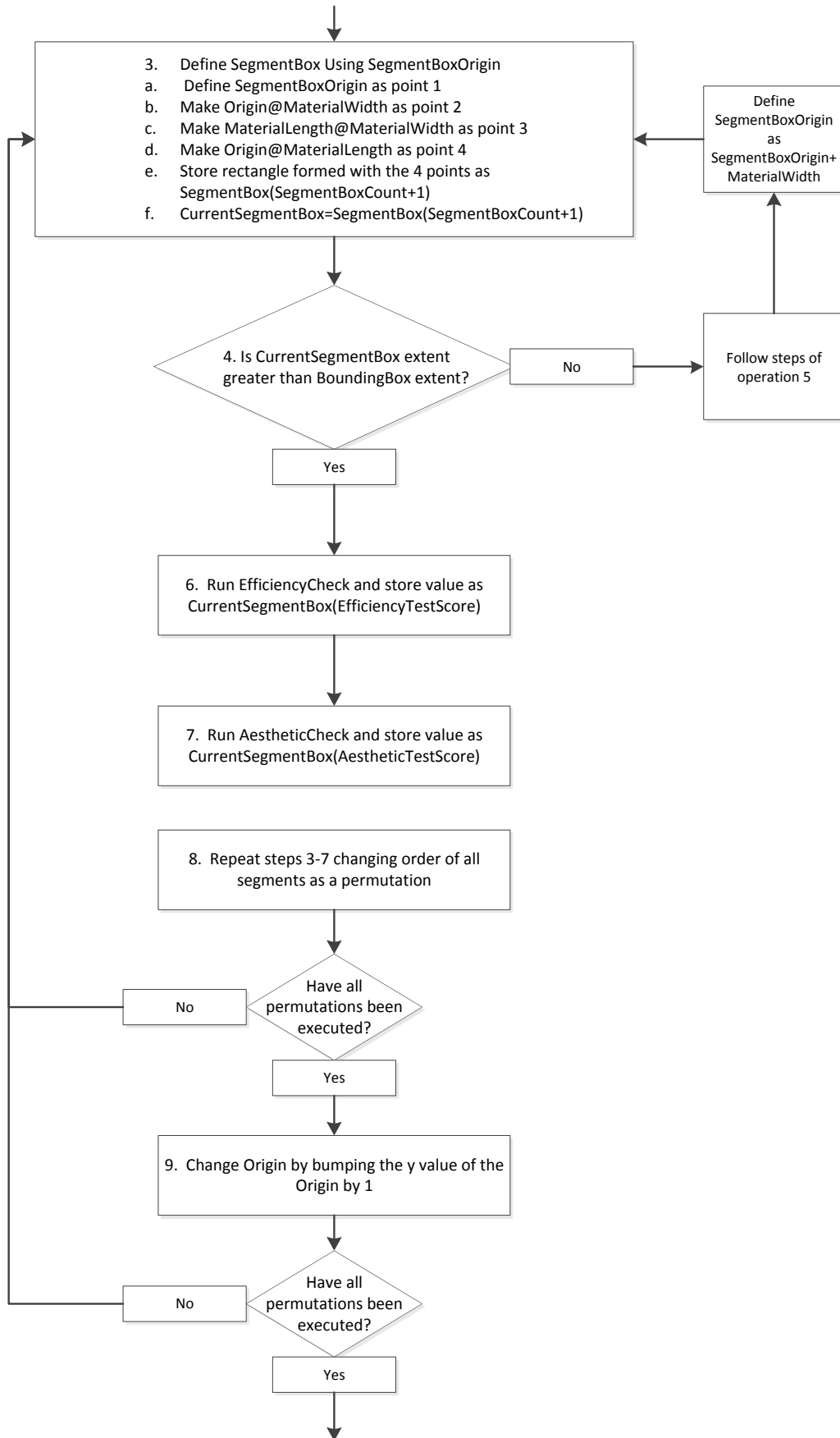
- a. Create a bounding box using the origin, furthest points on x and y axis, and their intersecting point
3. Define Segment Box
 - a. Origin is defined as point 1
 - b. From origin move greatest width of defined material on y axis
 - i. Store as point 2
 - c. Move to greatest value of bounding box in x value, remaining with $y = 12$
 - i. Store as point 3
 - d. Move, on same valued x-axis, down 12 on y axis
 - i. Store as point 4 and return to origin
 - e. Store rectangle formed as W_n
4. Continue process of segment box until the Bounding Box is fully divided
 - a. Use point 2 from previous segment box as new location of origin
 - b. Store as $W_{(n+1)}$
5. Intersect W_n with floor plan
 - a. Take the closest and furthest x value of newly formed shape
 - i. Store that in memory as ft. run of material used
 - b. Can material be placed without any obstructions?
 - i. If yes then;
 1. End function
 - ii. If no then;
 1. Check database for stored material that can be used
 - a. If yes then;
 - i. Use it
 - b. If no then;
 - i. Store dimension of material in database

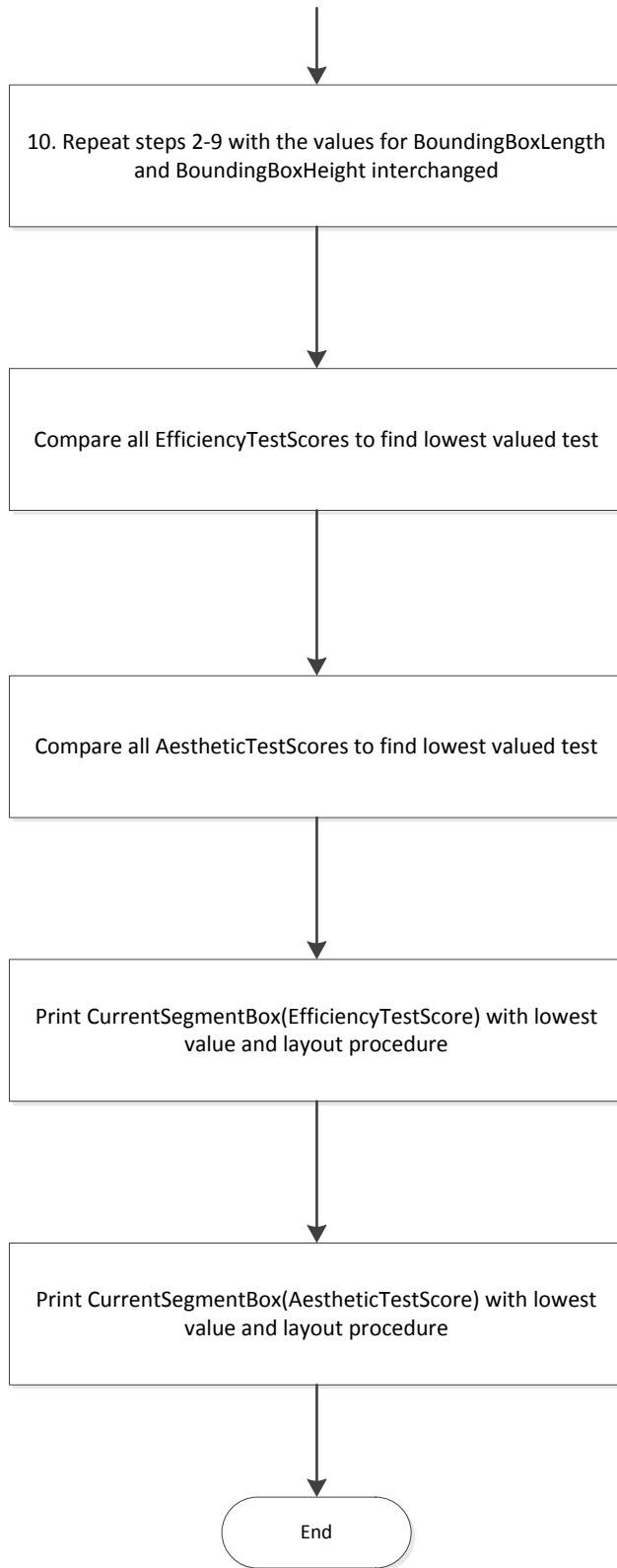
- c. Do the edge segments of the new shape share the same segments of the floor plan?
 - i. If yes then;
 1. End function;
 - ii. If no then;
 1. Store the length of that segment in memory
6. Repeat Process 5 with $W(n+1)$ until all segments complete
7. Run Efficiency Check
 - a. Store value in memory as $Test_{n-1}$
8. Run Aesthetic Check
 - a. Store value in memory as $Test_{n-1}$
9. Repeat steps 5-8 changing order of all segments as a permutation
 - a. Store value in memory as $Test(n+1)$
10. Change point of origin of $W1$ by bumping the y value by 1
 - a. Repeat steps 3-8
 - b. Store value in memory as $Test(n+1)$
11. Repeat steps 1-10 with the values for x and y interchanged
 - a. Store value in memory as $Test(n+1)$
12. Compare all Efficiency Check Tests to find the lowest valued $Test_n$
13. Compare all Aesthetic Check Tests to find the lowest valued $Test_n$
14. Print $Test_n$ with lowest valued Efficiency Check and layout procedure
15. Print $Test_n$ with lowest valued Aesthetic Check and layout procedure

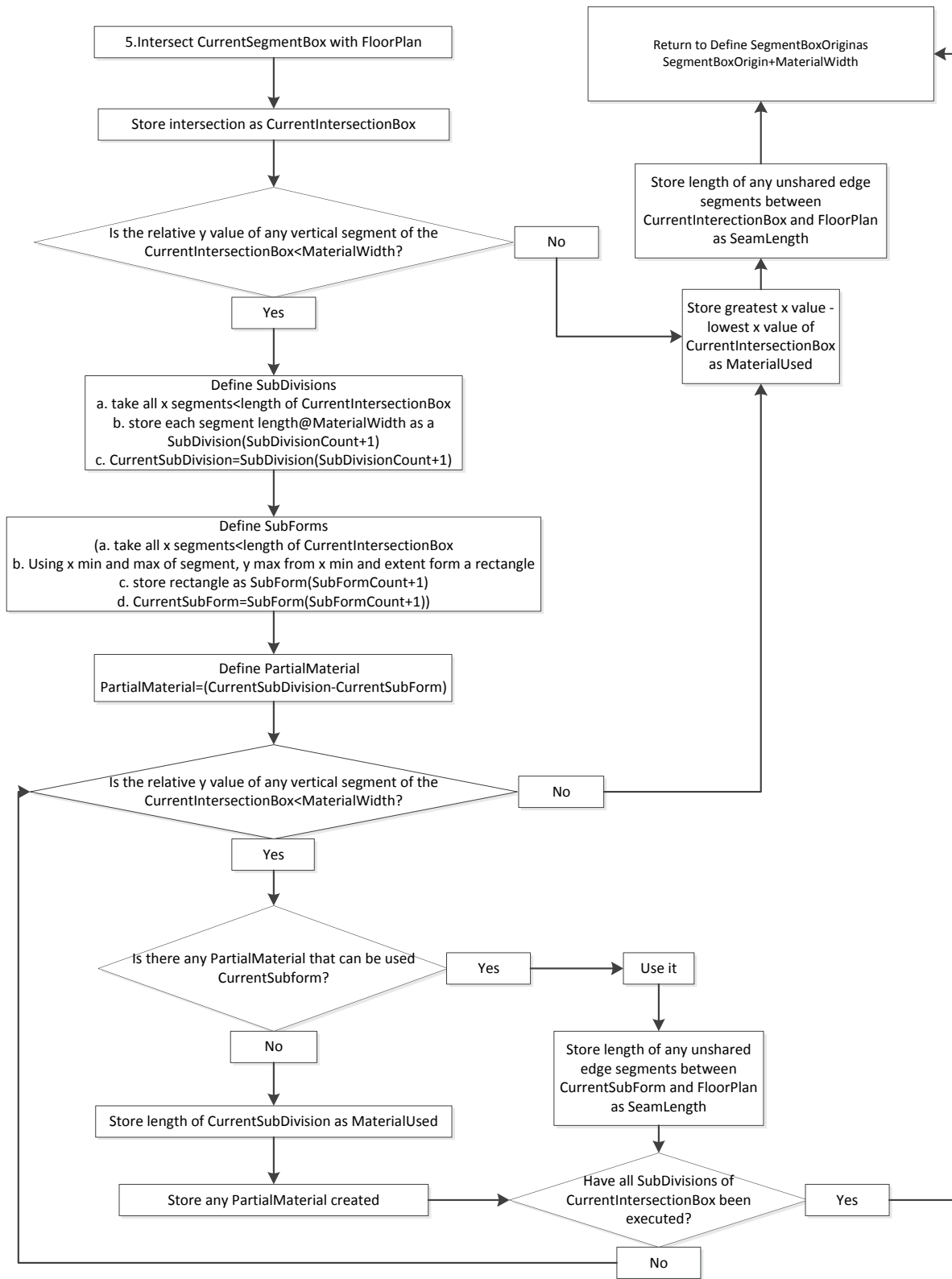
End

After the completing of the psuedocode, the next step was creating a flowchart of the operations and defining the variables more specifically.









Expanded View of Operation 5

The flowchart was created with Microsoft Visio, which has an interface that is well adapted for flowchart creation. The descriptive images following the flowcharts were constructed with the use of Adobe's Photoshop and InDesign. Figures 5-8 are visual representations of the operations and variables used in the Operation 5 of the flowchart for better clarity and understanding.

Figure 5

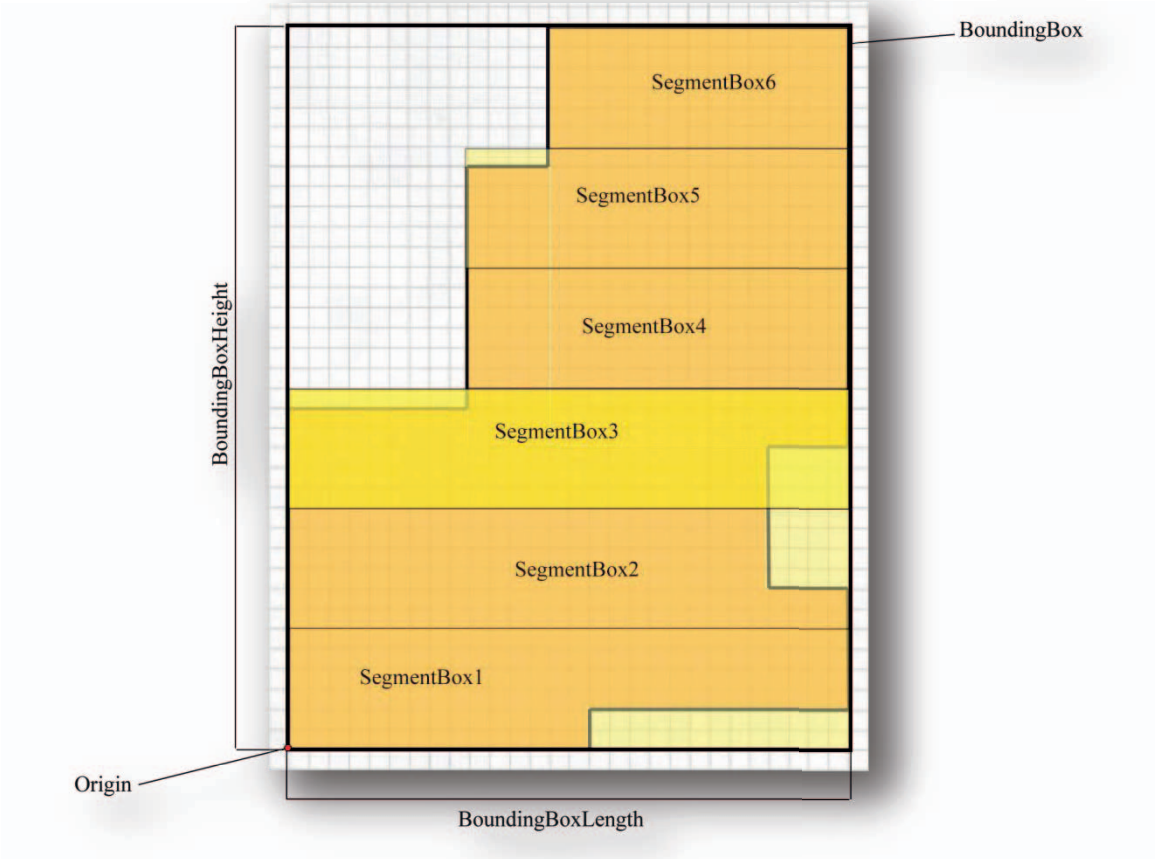


Figure 5 is a representation of the floor plan with the Origin, BoundingBox, and SegmentBoxes overlaid and defined.

Figure 6

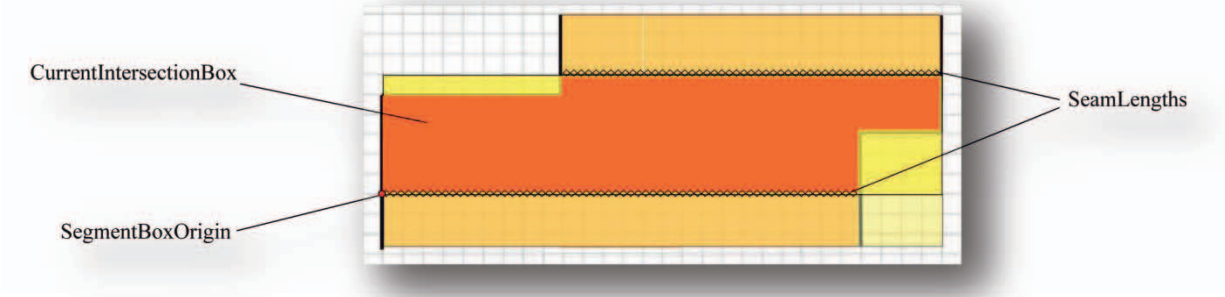


Figure 6 defines the CurrentIntersectionBox, SegmentBoxOrigin, and SeamLengths

Figure 7

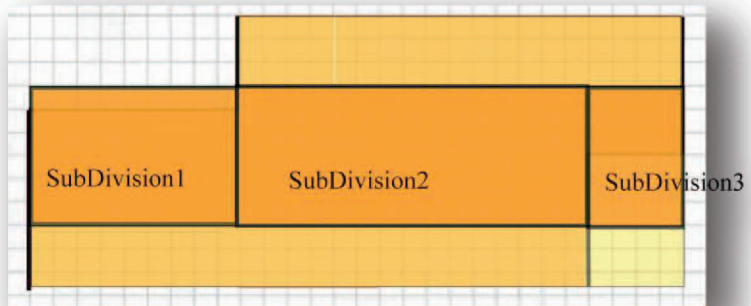


Figure 7 shows the SubDivisions defined by the in the length of the x values that are less than the full length of the CurrentIntersectionBox and assigning each with a number.

Figure 8

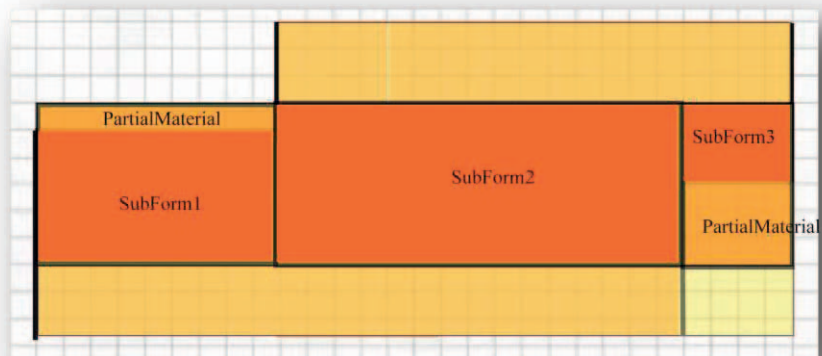


Figure 8 defines the PartialMaterial and SubForms produced from the SubDivisions

As of now this process will allow a user to analyze a floor plan and determine the most efficient layout of material and also a more aesthetic layout. The aesthetic value is based off the idea of having less seam lines (the joining of two pieces of material) holds a potential higher level visual quality. After completing the decision to try and push this idea forth in a personal effort, the next step became writing the computer program. The following is the Visual Basic code that was created in Visual Studio to produce the resultant program.

```
Imports System.Drawing
Imports System.Drawing.Drawing2D
Imports System.Windows.Forms

Public Class Form1

    Private Sub Form1_Load(ByVal sender As System.Object, ByVal e As System.EventArgs)
Handles MyBase.Load

    End Sub

    Protected Overrides Sub OnPaint(ByVal e As PaintEventArgs)

        Dim g As Graphics = e.Graphics
        g.Clear(Me.BackColor)

        Dim pen As New Pen(Color.Blue, 3)
        Dim brush As New SolidBrush(Color.Red)
        Dim brush2 As New SolidBrush(Color.Blue)

        Dim polygonPointsInput As String
        polygonPointsInput = InputBox("Enter number of points in polygon")

        Dim PolygonPointCount As Integer
        PolygonPointCount = (Val(polygonPointsInput))

        Dim PointArrayCount As Integer = PolygonPointCount - 1
```



```

Dim points(PointArrayCount) As PointF

Dim PointCounter As Integer = 0
While PointCounter < PolygonPointCount
    Dim p As PointF
    Dim x As String
    x = InputBox("Enter x coordinate of point " & (PointCounter + 1))
    Dim y As String
    y = InputBox("Enter y coordinate of point " & (PointCounter + 1))
    p.X = (Val(x))
    p.Y = (Val(y))
    points(PointCounter) = (p)
    PointCounter = PointCounter + 1
End While

Dim theFloorPlan As Drawing2D.GraphicsPath
theFloorPlan = New Drawing2D.GraphicsPath
theFloorPlan.Reset()
theFloorPlan.AddPolygon(points)
Dim floorPlanRegion As Region = New Region(theFloorPlan)
Dim floorPlanBoundsRect As RectangleF = floorPlanRegion.GetBounds(g)

Dim CarpetWidth As Integer
Dim CarpetLength As Integer
Dim carpetWidthInput As String
carpetWidthInput = InputBox("Enter width of carpet roll")
CarpetWidth = (Val(carpetWidthInput))
Dim carpetLengthInput As String
carpetLengthInput = InputBox("Enter length of carpet roll")

```

```

CarpetLength = (Val(carpetLengthInput))

Dim carpetStartPoint As Point
Dim r As String
r = InputBox("Enter x coordinate of carpet start point")
Dim s As String
s = InputBox("Enter y coordinate of carpet start point")
carpetStartPoint.X = (Val(r))
carpetStartPoint.Y = (Val(s))

Dim carpetSize As Size
carpetSize = New Size(CarpetLength, CarpetWidth)

Dim regionRectArea As Double
Dim wastedArea As Double
wastedArea = 0

Dim carpetRegionLoops As Integer = Decimal.Ceiling(floorPlanBoundsRect.Height /
CarpetWidth)
Dim loopCounter As Integer

For loopCounter = 1 To carpetRegionLoops

    Dim carpetRegion As Rectangle
    carpetRegion = New Rectangle(carpetStartPoint, carpetSize)
    Dim region As Region = New Region(carpetRegion)
    region.Intersect(floorPlanRegion)
    Dim region2 As Region = New Region(carpetRegion)
    region.Xor(region2)

```

```

Dim transformMatrix As Matrix
transformMatrix = New Matrix

For Each regionRect As RectangleF In region.GetRegionScans(transformMatrix)
    regionRectArea = regionRect.Width * regionRect.Height
    wastedArea = wastedArea + regionRectArea
Next

carpetStartPoint.Y = (carpetStartPoint.Y) + CarpetWidth
CarpetWidth = CarpetWidth + CarpetWidth

Next

MsgBox("The accumulated wastage of carpet is " & wastedArea & " sq. ft.")

g.TranslateTransform(10.0F, 10.0F)
g.FillRegion(brush, floorPlanRegion)
brush.Dispose()

End Sub

Public Sub New()
    MyBase.New()
    InitializeComponent()
    Me.AutoScaleBaseSize = New System.Drawing.Size(5, 5)
    Me.ClientSize = New System.Drawing.Size(400, 400)
    Me.StartPosition = System.Windows.Forms.FormStartPosition.CenterScreen
End Sub

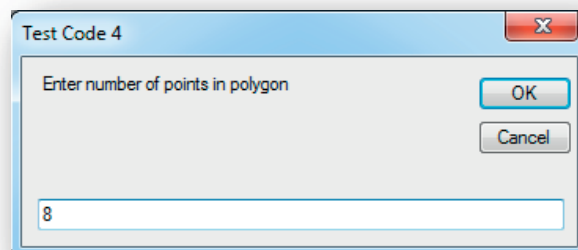
End Class

```

Conclusion of Results

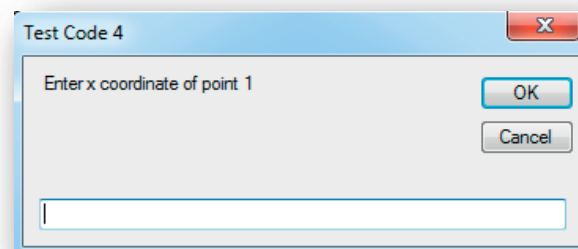
The resulting program sends you through a series of prompt windows which asks for the data input for the design at hand. It will start by asking the total number of points for the floor plan, or polygon parameters, then the specific x and y values of each point. After processing the form the program begins asking about the material's width, length, and the starting coordinate. This program then returns the square footage of material that is declared waste. In this program waste isn't calculated as accurately as it is presented in the flow chart. At this point in the program's evolution, the partial material is not directed into any reusable material data memory bank, thus any partial material is being declared as waste. The following figures present the program's input inquiry and data output.

Figure 9 - *This is the initial window that is asking for the number of points in the polygon.*



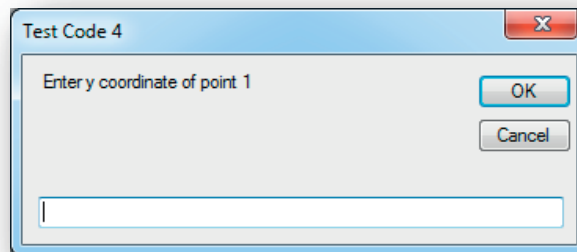
A screenshot of a Windows-style dialog box titled "Test Code 4". The dialog has a light blue header bar with a close button (X) in the top right corner. The main area is white and contains the text "Enter number of points in polygon" followed by "OK" and "Cancel" buttons. Below the text is a text input field containing the number "8".

Figure 10 - *Next the program is going to begin asking for the x coordinates for each point.*



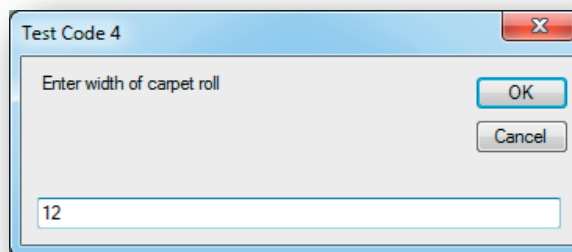
A screenshot of a Windows-style dialog box titled "Test Code 4". The dialog has a light blue header bar with a close button (X) in the top right corner. The main area is white and contains the text "Enter x coordinate of point 1" followed by "OK" and "Cancel" buttons. Below the text is an empty text input field.

Figure 11 - With each x value a corresponding y coordinate will need to be assigned for each point.



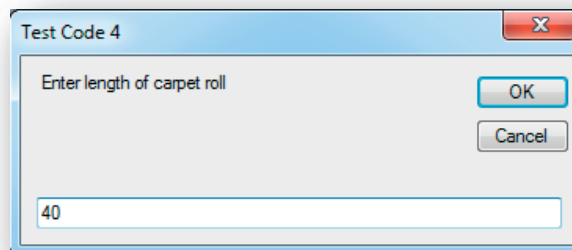
A screenshot of a dialog box titled "Test Code 4". The dialog has a red close button in the top right corner. The main text reads "Enter y coordinate of point 1". Below the text is a text input field that is currently empty. To the right of the input field are two buttons: "OK" and "Cancel".

Figure 12 - Now all coordinates have been assigned and the program is asking you to define the materials width.



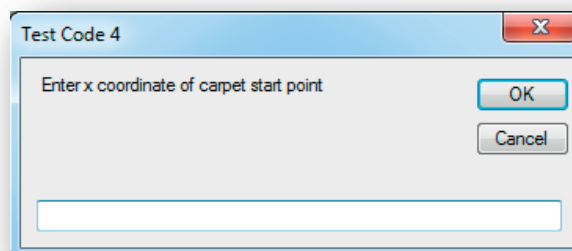
A screenshot of a dialog box titled "Test Code 4". The dialog has a red close button in the top right corner. The main text reads "Enter width of carpet roll". Below the text is a text input field containing the number "12". To the right of the input field are two buttons: "OK" and "Cancel".

Figure 13 - The next window is asking for a defined length of the material. The length is the same as the greatest value of the bounding box,



A screenshot of a dialog box titled "Test Code 4". The dialog has a red close button in the top right corner. The main text reads "Enter length of carpet roll". Below the text is a text input field containing the number "40". To the right of the input field are two buttons: "OK" and "Cancel".

Figure 14 - The program now needs an origin point. This again is defined by the coordinates being entered.



A screenshot of a dialog box titled "Test Code 4". The dialog has a red close button in the top right corner. The main text reads "Enter x coordinate of carpet start point". Below the text is a text input field that is currently empty. To the right of the input field are two buttons: "OK" and "Cancel".

Figure 15 – *After the x coordinate is entered the y coordinate is requested.*

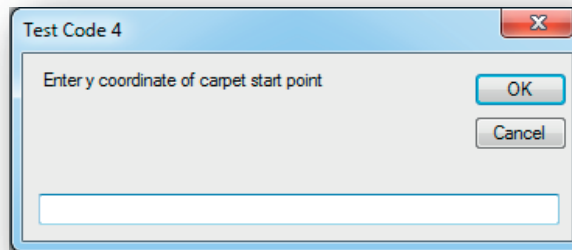
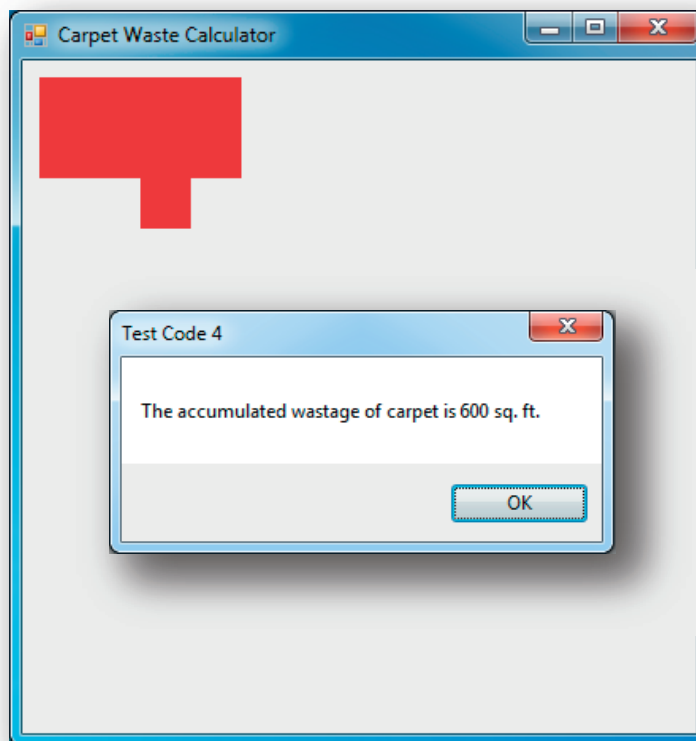


Figure 16 – *Once all the requested information has been inputted the program goes to work. This figure shows the how the program returns the resulting information both quantitatively and as a visual representation of the polygon that was defined.*



Conclusion and Projection

In conclusion this research provides as a stepping stone toward the ultimate achievement, a tool that will calculate a building's waste from a particular design. Taking this program forth requires applying the same steps and procedures toward each individual component of a building system. Beyond producing the sum of the waste for a particular design, this program would allow for a comparative waste analysis. One would be able to analyze how a design that would reduce the amount of carpet or flooring material is going to affect the amount of waste for the drywall and 2x4's for a wall component. Further research and development on this program would help aid the advancement of BIM tools and their need in efficient management of resources during the building design process.

References

Kalay, Yehuda E. (1989). *Modeling objects and environments*. New York: John Wiley & Sons.

Knuth, Donald E. (1968). *Fundamental Algorithms: the art of computer programming*. Reading, MA: Addison-Wesley Publishing Company.

Stephens, Rod (2000). *Visual Basic Graphic Programming, Second Edition: hands-on applications and advanced color development*. New York: John Wiley & Sons.