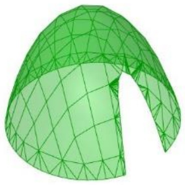
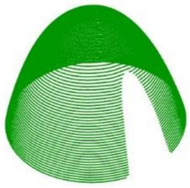
Pseudocode for bricklaying (earth bag) on domes (Grasshopper + Anemone + Weaverbird)

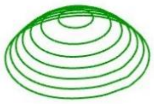
```
#input the mesh with door openings and arched connection
```



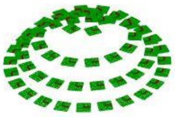
```
# make contours on the surface of the mesh, the height of the brick/2 is the contour interval (this process gives twice the number of contours needed)
```



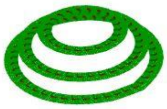
```
# cull the contours at odd numbers i.e. first, third, fifth, seventh…
# divide the remaining contours into two sets based on the closed curves and open curves.
# brick laying on closed curves:
```



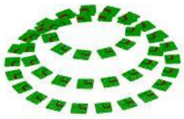
```
        #further divide the contours with respect to alternate courses
        #divide by length, first set of contours, length of division = length of brick + mortar gap
        #create horizontal frames at each point for these contours
```



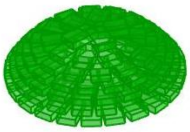
```
        #divide by length, second set of alternating contours, length of division = length of brick + mortar gap
        #create twice the number of horizontal frames along these curves
```



```
        #cull the first, third, fifth……and all odd number horizontal frames
        #list the final even number horizontal frames
```



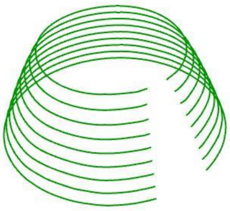
#generate box using the box component, input the horizontal frames from both sets into planes and enter the size of the brick in x, y and z direction



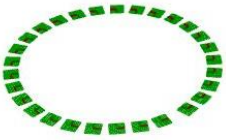
#brick laying on open curves:

#Clean the data tree of the contours for all null and invalid items.
#further divide the contours with respect to alternate courses
#sort list for each set
#list length for each set (this list length is the input for the number of repeats in a loop.)
# each curve is brought into the loop as a separate branch

#First set

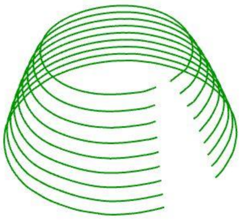


#divide the length of first set of each curve, length of division = length of brick + mortar gap
#create horizontal frames at each point for these contours



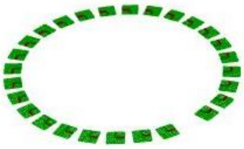
#close the loop
#toggle button and repeat the task for all the number of curves in the set (repeat input from list length)

#Second set



#divide by length, second set of alternating contours, length of division = length of brick + mortar gap
#create twice the number of horizontal frames along these curves

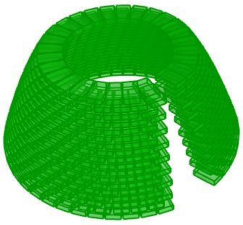
#cull the first, third, fifth……and all odd number horizontal frames



#list the final even number horizontal frames
#close the loop
#toggle button and repeat the task for all the number of curves in the set (repeat input from list length)

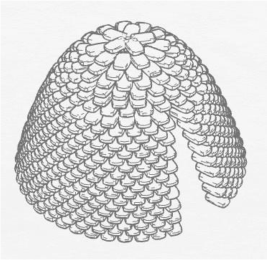
#generate box using the box component, input the horizontal frames from both sets into planes and enter the size of the brick in x, y and z direction.

#merge the bricks generated on the closed and open curves



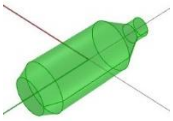
#convert these breps (bricks) into mesh

#smoothen mesh using weaverbirds Laplacian smoothening to generate earth bags (smoothened bricks)

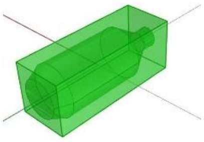


Pseudocode for laying bottle on domes (Grasshopper + Anemone + Weaverbird)

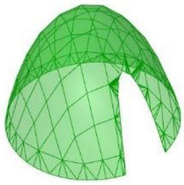
#model a bottle in Rhino and reference it as a brep in grasshopper (align the bottle in the xy axis and the centroid of the bottle is located at 0,0,0)



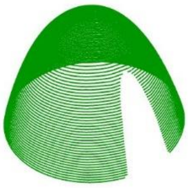
```
#create a bounding box for the bottle
```



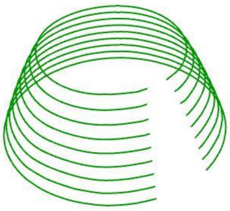
```
#identify the x and z dimensions of the bottle by deconstructing the bounding box (this z dimension determines the contour heights)
#input the mesh with door openings and arched connection
```



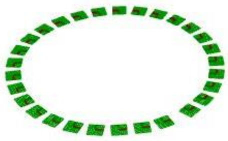
```
# make contours on the surface of the mesh, the height of the bottle/2 is the contour interval (this process gives twice the number of contours needed)
```



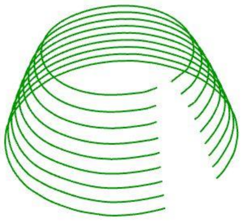
```
#Clean the data tree of the contours for all null and invalid items.
# cull the contours at odd numbers i.e. first, third, fifth, seventh…
#further divide the contours with respect to alternate courses
#sort list for each set
#list length for each set (this list length is the input for the number of repeats in a loop.)
# each curve is brought into the loop as a separate branch
        #First set
```



```
        #divide the length of first set of each curve, length of division = length of brick + mortar gap
        #create horizontal frames at each point for these contours
```

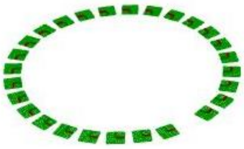


```
    #close the loop
    #toggle button and repeat the task for all the number of curves in the set (repeat input from list length)

    #Second set
```

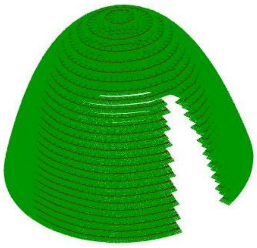


```
    #divide by length, second set of alternating contours, length of division = length of brick + mortar gap
    #create twice the number of horizontal frames along these curves
```

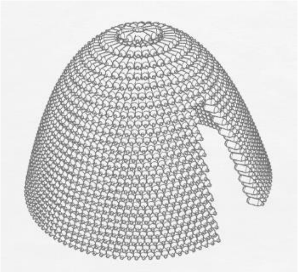
```
    #cull the first, third, fifth……and all odd number horizontal frames
```



```
    #list the final even number horizontal frames
    #close the loop
    #toggle button and repeat the task for all the number of curves in the set (repeat input from list length)
# merge all the frames
```



```
#orient the bottle to each of the horizontal frames

    #the plane of the bounding box of the bottle is the source and the horizontal frames are the target for the bottle geometry.
```



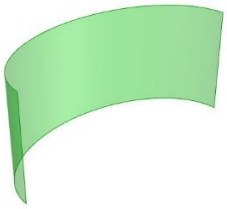
Pseudocode for laying bricklaying on walls (Grasshopper)

#description

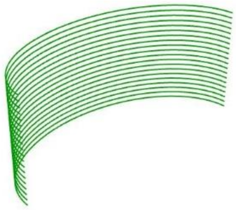
> The brick pattern is divided into pixels on the surface geometry for identification of the pattern. The size of the pixel is derived from the size of the brick. The combination of these pixels thus generate patterns. Also, the combined sizes of the pixels are either a whole brick or half the size of the brick, in other words, two brick sizes are used to generate the patterns: a whole brick and half the size of the brick.

#set the dimensions (L x B x H) of the brick

#reference the surface as brep for bricklaying in grasshopper



#divide the contour on the surface, the contour height is the height of the bricks

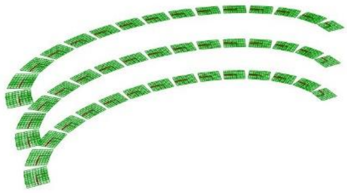


#cull the first contour (from bottom) in the list

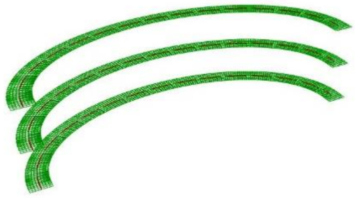
#Pattern 1

- **6 course pattern i.e. pattern repeats after every 6th course**
- **pixel length = ¼ of length of full brick**
- **two sizes of bricks used in the composition**

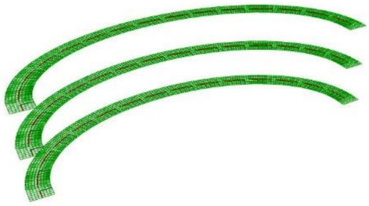
```
#divide the length of every sixth curve (starting from the first) by ¼ length of the brick,
create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
True True  False  False  True  True
```



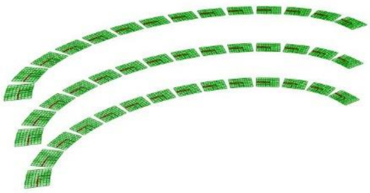
```
#divide the length of every sixth curve (starting from the second) by ¼ length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
True False False False False True
```



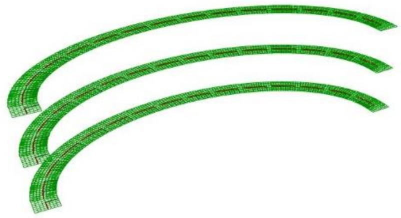
```
#divide the length of every sixth curve (starting from the third) by ¼ length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
False False True True False False
```



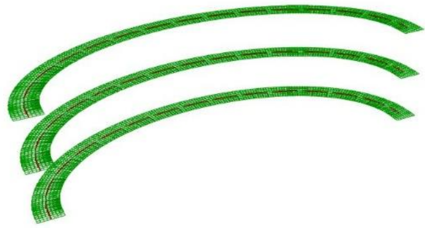
```
#divide the length of every sixth curve (starting from the fourth) by ¼ length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
False True True True True False
```



```
#divide the length of every sixth curve (starting from the fifth) by ¼ length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
False True True True True False
```

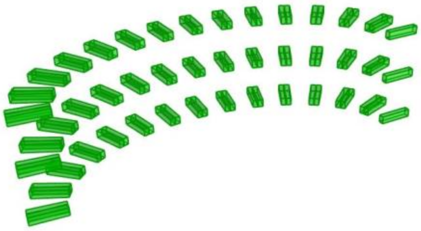


#divide the length of every sixth curve (starting from the sixth) by ¼ length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
True False False False False True

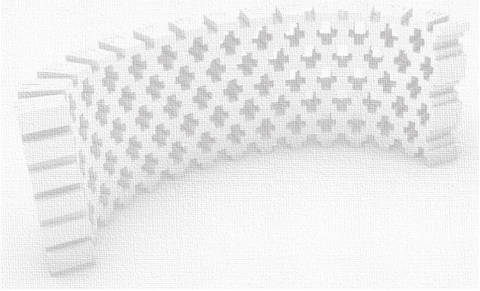


#create boxes (pixel only in elevation) at each of these frames, box (pixel) dimension:
L = ¼ length of brick
B = breadth of the brick
H = height of the brick

#Create a solid union of these boxes individually for each course



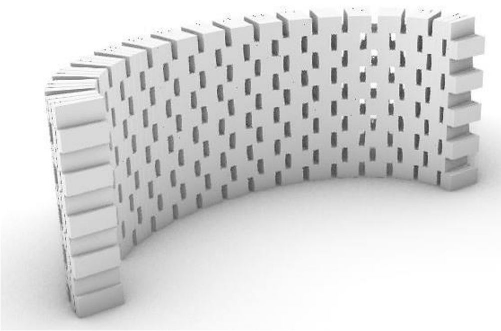
#merge all the data for the boxes to get the result of brick pattern.



#Pattern 2

- **4 course pattern i.e. pattern repeats after every 4th course**
- **pixel length = 1/3 of length of full brick**
- **only full bricks used in the composition**

```
#divide the length of every fourth curve (starting from the first) by 1/3 length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
True True False True

#divide the length of every fourth curve (starting from the second) by 1/3 length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
True True False True

#divide the length of every fourth curve (starting from the third) by 1/3 length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
False True True True

#divide the length of every fourth curve (starting from the fourth) by 1/3 length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
False True True True

#create boxes (pixel only in elevation) at each of these frames, box (pixel) dimension:
L = 1/3 length of brick
B = breadth of the brick
H = height of the brick

#Create a solid union of these boxes individually for each course
#merge all the data for the boxes to get the result of brick pattern.
```



#Pattern 3

- **2 course pattern i.e. pattern repeats after every 2th course**
- **pixel length = 1/6 of length of full brick**
- **only full bricks used in the composition**

```
#divide the length of every second curve (starting from the first) by 1/6 length of the brick, create a series to repeat the divide procedure.
```

```
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
True True True True False False

#divide the length of every second curve (starting from the second) by 1/6 length of the brick, create a series to repeat the divide procedure.
#generate horizontal frames at all these points of the divided curve
#cull pattern of the horizontal frame based on the brick course pattern
True False False True True True
```