Design Document

Global Variables

**Stones left to move** = integer

**Turn** = 1

**Pit Markers** = array

**Background** = background surface

**Extra\_turn** = false

**Pits** = array

**Game\_over** = false

Classes

**Cursor (Sprite)**

**Variables:**

Player = 1

Image, rect = Loaded image and rect

**Functions:**

*Update:*

Moves to mouse position. Changes color based on current player.

**Pit (Sprite)**

**Variables:**

Pit Counter = A Pit Marker

Sister Pit = integer

count = 3

controller = 1

Khalana = False

Image, Rect = Loaded image and rect

Opposite Pit = integer

**Functions:**

*Chosen (if not Khalana)*

Triggered when clicked, if turn is = controller of pit counter and pit counter doesn’t have no stones in it

Sets count to 0, sends a stone to its sister pit, setting global variable stones left to previous count

If extra\_turn == true:

Set extra\_turn to false.

Else:

Set player to 2-current player + 1 (next turn).

*Update*

Shows the count as the number on top of itself

If the number is 0, it simply removes the image of the number

Changes image to long oval if khalana

*Add*

If the stones left to move != 1:

Add 1 to count.

If the stones left to move == 1:

If Khalana, set extra\_turn to true.

If count is 1, set the count of pits[opposite\_pit] to 0 and add the previous number that it had to the khalana with controller = turn.

Main()

Initializes everything imaged-based (background, sprites, draws images)

Creates 6 pit sprites for each player and positions them

Creates 1 pit sprite with khalana=true for each player and positions them

**Main Loop:**

Every 60 ticks of the pygame clock:

Check for quitting

Check if all counts of pits on one side are 0. If so, move all stones from all non-khalana pits into their controller’s khalana. Then print the winner on the screen and set game\_over to true.

Update all the sprites. Update background. Draw sprites to background. Flip screen.

If game\_over is true, wait 1 second, then quit.