Physics todo:

* Make collision points a dequeue, take points out of the front when full (and also make it inside a collision callback function)
* Remove polygon normals, they’re not used (alternatively: remember why I calculated normals)
* Change polygon pointcase to see if the point is to the left of all lines, if so, return true
* Make step forward account for iteration count
* Add compound collision properly (also add modifier tool to bind two colliders together)
* Make physics part a library + a different project in the solution
* Add collision layers
* Remove shape destructor stuff
* Make pointcast for planes work without interfering with modifier tools
* Add collision callback
* Make grab more powerful
* Fix collision points not working with rotated shapes
* Make impulses and position movings apply after everything has been calculated, (not calculate one collision, apply one change; instead calculate all collisions, apply all changes)
* Stop buttons from being reset on clear
* Centre colliders around local (0,0) for rotation purposes
* Add disable gravity button
* Add make static modifier tool
* Add shape tool for user to create a polygon from a set of points (convex hull creator is already part of the project)
* Make modifying radius while creating shape actually modify the shape radius
* Add polygon-polygon and polygon-cylinder and polygon-plane collision point getters
* Convert plane function into an epa function (just for science, it’s probably slower though so convert it back afterward)
* Put GJK and EPA functions into the CollisionManager class
* Make scrolling when creating polygon add polygon points
* Make slider automatically scale text and have a label
* Add UI for:
  + Modifying amount of points on polygon
  + Modifying grid size
  + Changing launch force
  + Make object static toggle
  + Window surrounding tools, also Minimising/maximising tool window

Links to look at:

[dyn4j](https://dyn4j.org/2021/05/2021-05-02-a-faster-broadphase-in-dyn4j-4-0-0/)

^-- broadphase stuff

<http://allenchou.net/2013/12/game-physics-contact-generation-epa/>

^-- this one looks real interesting, specifically the contact data stuff (although it is about 3D collision stuff)

<https://dyn4j.org/2010/04/gjk-distance-closest-points/>

<https://dyn4j.org/2010/05/epa-expanding-polytope-algorithm/>

\/ has the equation for impulse with torque and how it is derived

<https://en.wikipedia.org/wiki/Collision_response>

<https://erikonarheim.com/posts/understanding-collision-constraint-solvers/>

<https://www.youtube.com/watch?v=6rgiPrzqt9w>

[Collision Detection](http://www.jeffreythompson.org/collision-detection/line-circle.php)

[Extracting face/hit data after a GJK step - Math and Physics - GameDev.net](https://gamedev.net/forums/topic/693456-extracting-facehit-data-after-a-gjk-step/5362656/)

[PowerPoint Presentation](http://media.steampowered.com/apps/valve/2015/DirkGregorius_Contacts.pdf)

[computational geometry - How to resolve collisions of compound shapes using SAT? - Game Development Stack Exchange](https://gamedev.stackexchange.com/questions/17180/how-to-resolve-collisions-of-compound-shapes-using-sat)

[game physics - Calculate moment of inertia given an arbitrary convex 2D polygon - Stack Overflow](https://stackoverflow.com/questions/31106438/calculate-moment-of-inertia-given-an-arbitrary-convex-2d-polygon)

when making a game - <https://aie.instructure.com/courses/811/pages/Physics%20for%20Games%20-%20Making%20a%20Physics%20Game?titleize=0>