Graphics Pipeline

Want to convert:
1. objects defined as vertices of drawing primitives, and
2. viewing location and direction

into a discrete, rasterized representation of the objects (scene) seen from the view location (camera)

Graphics pipeline converts user-defined scene into rasterized image.

Basic order is:

vertices
↓
Transform → Clip → Project → Rasterize
↓

A more detailed description of the pipeline would be:
vertices
↓

[ modelling ] transformations → trans, rot, scale
→ 3D WC of objects into WC +
→ 3D WC of vertex & poly lighting

Transform
→ align WC axes with R·T(-VRP)
→ viewplane H, VP, X
→ view ref coords

Convert view frustum to standard form
→ normalized proj. coords

Clipping
→ clip against 3D view volume

Project
→ project onto 3D viewplane
→ viewplane position

Rasterize
→ rasterize into 2D viewport
→ sign conversion, depth buffering (Phong lighting)
→ pixels
Transformations:

Initial step in pipeline, used to position objects, lights, and virtual camera (view location and direction) relative to one another in a common world coordinate (WC) system.

Clipping:

Normally only a finite region of space can be seen by the camera (depends on camera's location, direction, and field-of-view angles). Anything outside this volume can be clipped away and ignored at later stages of the pipeline.
Near and far clip planes are normally inserted to define a finite viewing volume.

**Projection:**

Projection converts 3D objects into a 2D representation focused on a projection plane.

For perspective projection, rays are cast from camera center through each vertex in the view volume. Location ray intersects projection plane is that vertex's projected position.
For parallel projection, a set of rays are cast from each vertex in a common direction (i.e., rays are parallel to one another). Location ray intersects projection plane at that ray's projected position.

Rasterization:

Objects on projection plane are ordered relative to one another (depth buffering), converted into a discrete grid of samples (pixels) whose size matches the size of the viewport (window) into which they will be displayed.
Note shape (aspect ratio) of viewport can be different from shape of projection plane.