--:! {mine: []: (😃} <- **Mine Command Line Library** -> muse/docs/lib/mine.md
--😐 mine: *CLL operates plan to manage mines: shaft, bore, move to posts (between and within levels), and get ores.* -> mine, _mine

--:# Interface for mining plans: digging and navigating shafts and bores, extracting ore

--:# Plan elements for digging shafts going down, furnishing them going back up, and moving between levels.

--:> shafts: *Dig* -> `{shafts.name: ":", shafts.down: downs, shafts.back: levels, shafts.lower: levels, shafts.higher: levels}`

--:> shafts.name: *Included in error and status reports* -> `":"`

--:> shafts.down: *Plan for digging shaft one level through even and odd levels* -> `downs`

--:> shafts.back: *Plan for returning to minehead, placing fixtures and safety shelves for player* -> `levels`

--:> shafts.lower: *Plan for navigating from one (even or odd) level to the next lower* -> `levels`

--:> shafts.higher: *Plan for navigating from one (even or odd) level to the next higher* -> `levels`

--:# Plan definitions for digging shafts down from even to odd and from odd to even levels

--:> downs: *Mark, dig, ladder, and fill one level (for landing platforms)* -> `{downs.even: plan, downs.odd: plan, downs.last: plan }`

--:> downs.even: _Mark, dig, ladder, and fill an even level (for landing platforms) -> `plan`

--:> downs.odd: *Mark, dig, ladder, and fill an odd level (for landing platforms)* -> `plan`

--:> downs.last: *Mark last level dug* -> `plan`

--:# Plan definitions for just moving between odd and even levels

--:> levels: *Traverse (and fill shelves for player safety in back plan) shaft* -> `{levels.even: plan, levels.odd: plan}`

--:> levels.even: _Plan for navigating from one even level to the next -> `plan`

--:> levels.odd: *Plan for navigating from one even level to the next* -> `plan`

--:# Plan elements for boring and furnishing horizontal access tunnels at odd and even levels for mining

--:> bores: *Horizontals* -> `{bores.name: ":", bores.post: mine.post, bores.even: plan, bores.odd: plan, bores.fix: plan }`

--:> bores.name: *Included in error and status reports* -> `":"`

--:> bores.even: *Plan for boring tunnels at even levels* -> `plan`

--:> bores.odd: *Plan for boring tunnels at odd levels* -> `plan`

--:> bores.fix: *Plan for fixtures: shaft area ladders and placing barrel* -> `plan`

--:# **Interface definition for implemented bore navigation functions in mines**

--:> mine.post: *Navigate shaft and bores to go to marker.* -> (markerName: ":", :bores:): `marking[]`

--:: mine.mark(:plan:, :marking:) -> *Make place name, report result.* -> `markerName: ":", label: ":", report: ":"`
--:+ *Called by `worker.execute` to make marker name and use it to add map point for navigation in mine.*
--:+ *Puts plan name value in marker (keyed by `"shaft"` or `"bore"`) so marker is enough for navigating in shaft or bore.*
--:+ *Marker place name formed as `head:level:base` or `head:base` or `head` with place labelled as `"outer"|"inner"|"shaft"`.*

--:# **Go to minehead, dig shaft and return to minehead if no errors.**
--:+ *Start dig down from minehead, finish by placing marker one level below target level, error raised for failure.*
--:+ *Add "shaft" and shaft plans name as key and value to shaft plans for inclusion in marker.*
--:+ *Execute shaft plans back to return to minehead from one below target level*
--:+ *Add "shaft" and shaft plans name as key and value to shaft plans for inclusion in marker*

--:# **Do the `shaft` operation for the command line interface.**

--:- shaft minehead levels shaftPlans -> *Dig down number of levels under named minehead place using specified plans.*

--:# **Do the `post` operation for the command line interface**.

--:- post marker borePlans? -> *Go to marker (and up 1 block) from current level with saved or specified plans.*

--:# **Bore horizontal tunnels and mine tunnels branching off the main tunnel for ore**
--:+ *Use shaft plans and bore plans to navigate to marker, bore horizontal tunnels using bore plans.*
--:+ *Add "bore" and bore plans name as key and value to bore plans for inclusion in marker.*
--:+ *Bores plans at even or odd level, potentially leaves place marks and torches.*

--:- bore marker borePlans -> *Dig horizontally from marker using saved or specified bore and shaft plans.*
--:+ *Use shaft plans and instantiated bore plans to go to marker and execute mining operations*

--:- ores marker borePlans? -> *Excavate ores from side tunnel near marker, return up 1 from marker.*

--:: mine.op(arguments: :[op: ":", placeName: ":", borePlansFileOrLevels: ":"|#:, shaftPlansFile: ":"]) -> *Dig.* -> `^:, ":", #: &:`
--:+ *Dig shaft; go to post at level; bore, mark, and torch; get ores. Markers hold saved plans.*