```
preprocess_repository = {
foreach(O1,O2) in repository
match_ontologies O1,O2
end
}

match_ontologies O1,O2 = {
mapping =NeonToolKit(O1,O2)
C1 = extract_concept_from (O1)
C2 = extract_concept_from (O2)
th = threshold(C1,C2)
if (th > ths) create_hypermatching_record Hr
store_in_ hypermatching_record(C1,C2,Hr)
end
}

Clean_hypermatching_record = {
Foreach hypermatching_record Hr
Present_to_expert
if (expert_set_valid_mapping = false) delete Hr from hypermatching_record
else
continue to Hr+1
end
}

Hypersubsummed = get_subsumed_with_hypermappings(C1,C2)

compute_hypermodule(Hr){

foreach Hr     Hyper_matching_record
          get_matching_concept (C1,C2)
          hypersubsummed = get_subsumed_with_hypermappings(C1,C2)
          hypersubsummer = get_subsumer_with_hypermappings(C1,C2)
if empty (hypersubsummed and hypersubsummer){
          mono_module_record (C1,C2, Hr)

                    else
                    store_in_multi_module_record (C1,C2, Hr)
endif
}
foreach (C1,C2) in multi_module_record (C1,C2, Hr)
get_all_concepts_in_multimodule_record(c)
if (C1 subsumes_some_c = true and C1 subsumed_some_c = false) {
          upper_bound = (C1, Hr)
          else
          if (C1 subsumes_some_c = false and C1 subsumed_some_c = true) {
          lower_bound = (C1, Hr)
          }
endif
}
set_hypermodules(upper_bound Hr, lower_bound Hr, multimodule_record)

}
end
```