Algorithm 1 Construct CTI_SyscallSynonymBase

Input SyscallList[] /* a list of 354 linux syscall names (1-D array) */
Output SynonymList[] /* a table presenting SynonymBase (n-D array) */

Initialize *SynonymList* ← []
Initialize *PairList* ← []
for *thisSyscall* in *SyscallList* **do**
/* Search for manual webpage for each system call */
/* Get the sections of manual webpage */
    manpage ← crawl_web_content(url, *this_syscall*)
    nameSector ← extract(*manpage*)
    descriptionSector ← extract(*manpage*)
end for

/* Extract and store the verb-object phrase pair from manual */
for *sector* in [*nameSector* , *descriptionSector*] **do**
    verbList ← find_verb_phrase_in_sentence(*sector*) // by DP
    objectList ← find_object_phrase_in_sentence(*sector*) // by expert
    pairs ← construct_pairs(*verbList*, *objectList, sector*)
    extend list *pairs* to *PairList*
end for

/* Construct CTI_SyscallSynonymBase */
Initialize *rowData* ← []
for (*syscall*, *verb*, *object*) in *PairList* **do**
    rowData.syscall ← *syscall*
    rowData.enVerb ← *verb*
    rowData.object ← *object*
    rowData.sentence ← *descriptionSent*
    // We assign entity and action type for every syscall
    rowData.entityType ← classify_object_entity(*syscall, object*)
    rowData.actionType ← classify_syscall(*syscall*)
    // Obtain word vector for the 'verb' in sentence
    rowData.wordvector ← bert_embedding(*descriptionSent*, *verb*)
    append *rowData* to *SynonymList*
end for
return *SynonymList*