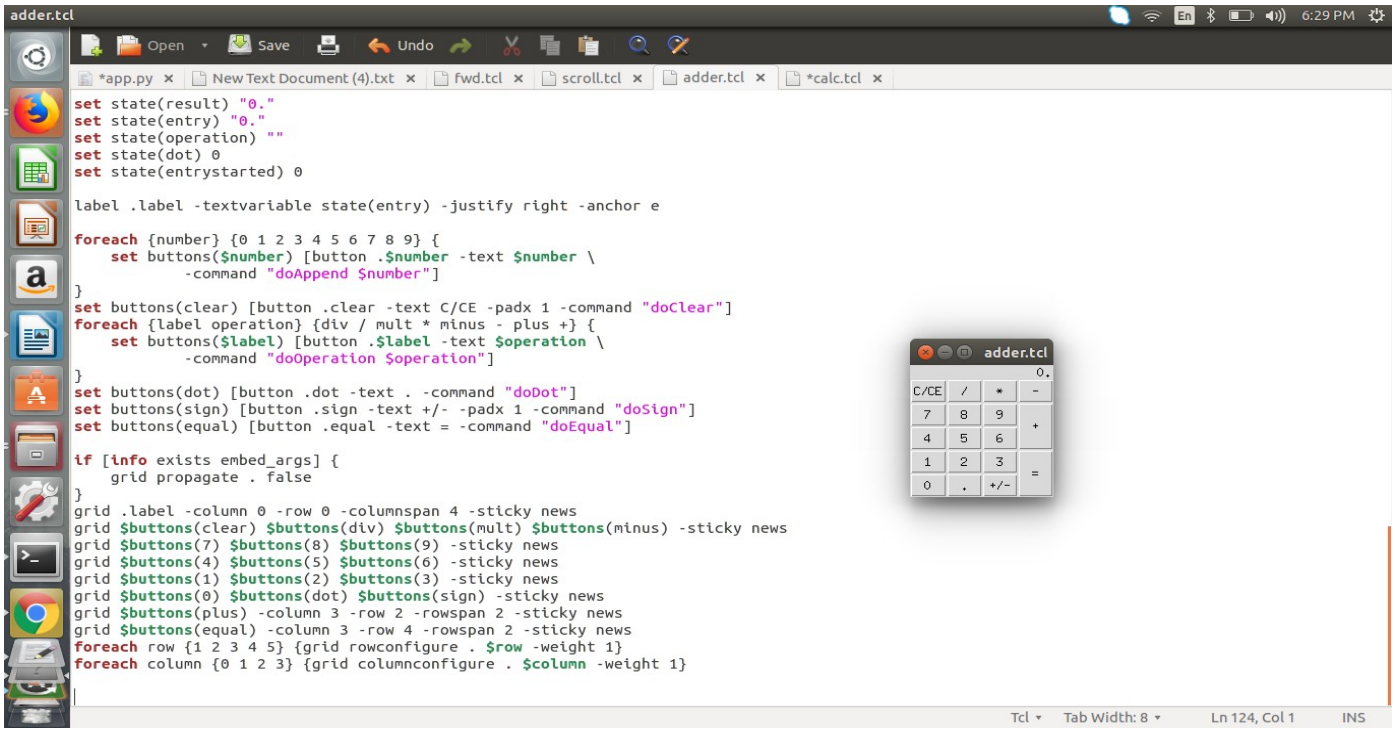
Assignment-7

Submitted By : Gargi Trivedi 2019PSP3023

Write a Tcl/Tk script to design a GUI for a basic calculator: add, sub, mul, and div.



Code :

```
proc doClear {} {
   global state

   set state(entry) "0."
   set state(dot) 0
   if {$state(entrystarted) == 0} {
         set state(result) "0."
         set state(operation) ""
   }
   set state(entrystarted) 0
}

proc doDot {} {
   global state

   set state(dot) 1
}
```

```
proc doAppend {what} {
   global state

   if {$state(entrystarted) == 0} {
         set state(entry) "0."
   }

   if {$state(operation) eq ""} {
         set state(result) "0."
   }

   if {!($what == 0 && $state(entry) eq "0.")} {
         set state(entrystarted) 1
      if {$state(dot) == 1} {
            set state(entry) [format "%s%s" $state(entry) $what]
         } else {
            regexp {([-0123456789]+).} $state(entry) foo integer

            if {$integer eq "0"} {
                  set state(entry) [format "%s." $what]
            } else {
                  set state(entry) [format "%s%s." $integer $what]
            }
         }
   }
}

proc doOperation {what} {
   global state

   if {$state(operation) ne ""} {
         doEqual
   }
   set state(operation) $what
   set state(result) $state(entry)
   set state(entrystarted) 0
   set state(dot) 0
}

proc doEqual {} {
   global state

   if {$state(operation) ne ""} {
         set state(result) \
               [expr "$state(result) $state(operation) $state(entry)"]
   } else {
         set state(result) $state(entry)
   }
   set state(entry) $state(result)
   set state(entrystarted) 0
   set state(operation) ""
   set state(dot) 0
```

```
}

proc doSign {} {
   global state

   if {$state(entry) ne "0."} {
         set sign ""
         set abs ""
         regexp {([-]?)([0-9.]+)} $state(entry) foo sign abs
      if {[string compare $sign "-"] == 0} {
            set state(entry) $abs
         } else {
            set state(entry) [format "-%s" $abs]
         }
         set state(entrystarted) 1
   }
}

set state(result) "0."
set state(entry) "0."
set state(operation) ""
set state(dot) 0
set state(entrystarted) 0

label .label -textvariable state(entry) -justify right -anchor e

foreach {number} {0 1 2 3 4 5 6 7 8 9} {
   set buttons($number) [button .$number -text $number \
            -command "doAppend $number"]
}
set buttons(clear) [button .clear -text C/CE -padx 1 -command "doClear"]
foreach {label operation} {div / mult * minus - plus +} {
   set buttons($label) [button .$label -text $operation \
            -command "doOperation $operation"]
}
set buttons(dot) [button .dot -text . -command "doDot"]
set buttons(sign) [button .sign -text +/- -padx 1 -command "doSign"]
set buttons(equal) [button .equal -text = -command "doEqual"]

if [info exists embed_args] {
   grid propagate . false
}
grid .label -column 0 -row 0 -columnspan 4 -sticky news
grid $buttons(clear) $buttons(div) $buttons(mult) $buttons(minus) -sticky news
grid $buttons(7) $buttons(8) $buttons(9) -sticky news
grid $buttons(4) $buttons(5) $buttons(6) -sticky news
grid $buttons(1) $buttons(2) $buttons(3) -sticky news
grid $buttons(0) $buttons(dot) $buttons(sign) -sticky news
grid $buttons(plus) -column 3 -row 2 -rowspan 2 -sticky news
grid $buttons(equal) -column 3 -row 4 -rowspan 2 -sticky news
foreach row {1 2 3 4 5} {grid rowconfigure . $row -weight 1}
foreach column {0 1 2 3} {grid columnconfigure . $column -weight 1}
```