

```

1 import tkinter.scrolledtext
2 from data import *
3 from process import process
4 from err import errCheckWithGui
5 from tkinter import filedialog
6 from tkinter import messagebox
7 import tkinter
8 import threading
9 from statusPanel import progressPanel
10 from statusPanel import errCheckProgressPanel
11 import sys
12
13 datas = None
14 winnerShown = False
15 panel:progressPanel = None
16
17
18 def initMainFunc():
19     global datas
20     filePath = filedialog.askopenfilename()
21
22     errPanel = errCheckProgressPanel()
23     errFlag = False
24     try:
25         with open(filePath,"rt",encoding="UTF-8") as blob:
26             errFlag = errCheckWithGui(blob,errPanel)
27             if(not errFlag):
28                 datas = data(blob)
29     except:
30         errPanel.root.quit()
31         messagebox.askyesno("err occured","during opening the
data file, err occured.")
32         errFlag = True
33
34     if(errFlag):
35         exit(0)
36
37
38 def getWinner(root:tkinter.Tk):
39     global datas,winnerShown,panel
40     if(winnerShown): return
41     else: winnerShown = True
42

```

```

43     def threadFunc(datas:data,panel:progressPanel):
44         winner = process(datas,panel)
45         if(winner != None):
46             tkinter.Label(root,text="The winner is " +
str(winner["id"])).pack()
47         else:
48             tkinter.Label(root,text="All candidates are
excluded!").pack()
49
50     processThread =
threading.Thread(target=threadFunc,args=(datas,panel))
51     processThread.start()
52
53
54
55 if __name__ == "__main__":
56     initMainFunc()
57     root = tkinter.Tk()
58
59     root.title("Vote system")
60     root.geometry("400x250")
61
62     commandList = tkinter.LabelFrame(root,text="command button")
63     commandList.pack(padx=10,pady=10)
64
65     start = tkinter.Button(commandList,text="start process")
66     start.bind("<1>",lambda event:getWinner(root))
67     start.pack(padx=10,pady=10,side=tkinter.LEFT)
68
69     end = tkinter.Button(commandList,text="exit")
70     end.bind("<1>",lambda event:root.destroy())
71     end.pack(padx=10,pady=10)
72
73     panel = progressPanel(root)
74     panel.showPanel()
75
76     root.mainloop()
77
78
79
80 from io import TextIOWrapper
81 import string
82 from multiprocessing import Pool
83 import os
84 import tkinter

```

```

85 from tkinter import ttk
86 from statusPanel import errCheckProgressPanel
87 import threading
88
89
90 # error check functions
91 # return:
92 # false -> no error found
93 # true -> error found
94
95 def wordCheck(text:str,panel:errCheckProgressPanel) -> bool:
96     panel.ProgressUpdate("wordCheck",{"progress":0})
97
98     if("CANDIDATES = " in text[:256] and
99         "VOTES = " in text[:256]): return False
100
101     panel.ProgressUpdate("wordCheck",{"progress":100})
102     return True
103
104 def valueCheck(text:str,panel:errCheckProgressPanel) -> bool:
105     panel.ProgressUpdate("valueCheck",{"status":
"start","progress":0})
106
107     maxLength = ("BFGHJKLMPQRUWXYZ-" +
string.ascii_lowercase).__len__()
108     for index,Achar in enumerate("BFGHJKLMPQRUWXYZ-" +
string.ascii_lowercase):
109         panel.ProgressUpdate("valueCheck",{"status":
"onProcess","index":index,"max":maxLength})
110         if(Achar in text): return True
111
112     panel.ProgressUpdate("valueCheck",{"status":
"end","progress":100})
113     return False
114
115 def checkVoteData(text:str,panel:errCheckProgressPanel) -> bool:
116     panel.ProgressUpdate("checkVoteData",{"status":
"start","progress":0})
117     lines = text.split("\n")
118
119     counter = 0
120     numberOfCandidate = 0
121     numberOfVote = 0
122     for line in lines:

```

```

123     element = line.split(" ")
124
125     if(counter == 0):
126         try:
127             numberOfCandidate = int(element[2])
128         except:
129             return True
130         counter += 1
131         continue
132
133     if(counter == 1):
134         try:
135             numberOfVote = int(element[2])
136         except:
137             return True
138         counter += 1
139         continue
140
141     if(counter == 2):
142         counter += 1
143         continue
144
145     votes = []
146     for i in range(numberOfCandidate):
147         try:
148             votes.append(int(element[i]))
149         except:
150             return True
151
152     checked = []
153     for index,aVote in enumerate(votes):
154         panel.ProgressUpdate("checkVoteData",{"status":
"onProcessVoteRow","index":index,"numberOfCandidate":numberOfCandidate})
155         for check in checked:
156             if(aVote == check): return True
157         checked.append(aVote)
158
159     if(counter == numberOfVote): break
160     panel.ProgressUpdate("checkVoteData",{"status":
"onProcessEntire","counter":counter,"numberOfVote":numberOfVote})
161     counter += 1
162
163     panel.ProgressUpdate("checkVoteData",{"status":

```

```

"end", "progress": 100})
164     return False
165
166
167     def
errCheckWithGui(blob: TextIOWrapper, panel: errCheckProgressPanel):
168         # panel = errCheckProgressPanel()
169         status = {"status": "None", "err": False}
170         check =
threading.Thread(target=errCehck, args=(blob, panel, status))
171         check.start()
172         panel.ProgressShow()
173         return status["err"]
174
175     # each error check should be separated into function
176     # and then executed as multiprocess
177     def
errCehck(blob: TextIOWrapper, panel: errCheckProgressPanel, status: dict):
178         text = blob.read()
179
180         # check include "CANDIDATES" and "VOTES" value
181         if(wordCheck(text, panel)):
182             status["err"] = True
183             status["status"] = "wordCheck"
184             panel.showErrorMessage()
185             panel.root.quit()
186             return
187
188         # check unwanted char are included
189         # check is there negative value
190         # check format correctness
191         if(valueCheck(text, panel)):
192             status["err"] = True
193             status["status"] = "valueCheck"
194             panel.showErrorMessage()
195             panel.root.quit()
196             return
197
198         # check duplicate data in vote data row
199         # check are there alphabets in vote data row
200         if(checkVoteData(text, panel)):
201             status["err"] = True
202             status["status"] = "checkVoteData"

```

```

203         panel.showErrorMessage()
204         panel.root.quit()
205         return
206
207     # check are there wrong amount of data in vote data row
208
209     blob.seek(0)
210     panel.root.destroy()
211     status["status"] = "end"
212     return False
213
214 import csv
215 from io import TextIOWrapper
216 import asyncio
217 import tkinter
218 from tkinter import ttk
219
220 class data:
221
222     def __init__(self, blob: TextIOWrapper) -> None:
223         self.candidateList = []
224         self.voteList = []
225
226         self.numberOfCandidate = 0
227         self.numberOfVote = 0
228
229         self.initProgressPanel()
230
231
232     threading.Thread(target=self.getData, args=(blob,)).start()
233         self.showProgress()
234
235     def initProgressPanel(self):
236         self.progressPanel = tkinter.Tk()
237         self.progressPanel.title("reading data...")
238
239         self.progressStatus = {
240             "readVote" :
tkinter.StringVar(self.progressPanel, "0"),
241             "currentData" :
tkinter.StringVar(self.progressPanel, ""),
242             "numberOfCandidate" :
tkinter.StringVar(self.progressPanel, "wait"),
243             "numberOfVote" :

```

```

tkinter.StringVar(self.progressPanel,"wait"),
243         "status"           :
tkinter.StringVar(self.progressPanel,"None")
244     }
245
246     self.progressBarState = {
247         "initStep"           :10,
248         "countVoteStepLength":50,
249         "initCandidateLength":40,
250
251     "prevStep"               :tkinter.IntVar(self.progressPanel,0),
252
253     "step"                   :tkinter.IntVar(self.progressPanel,0)
254     }
255     self.progressBar =
256     ttk.Progressbar(self.progressPanel,maximum=100,variable=self.progressBar
257     State["step"])
258
259     def showProgress(self) -> None:
260
261     tkinter.Label(self.progressPanel,textvariable=self.progressStatus["statu
262     s"]).pack()
263
264     frame = tkinter.Frame(self.progressPanel)
265     frame.pack()
266
267     #-----
268     infoPanel = tkinter.LabelFrame(frame,text="info")
269     infoPanel.pack(padx=50,pady=10,side=tkinter.RIGHT)
270
271     numberOfCandidate =
272     tkinter.LabelFrame(infoPanel,text="number of candidate")
273     numberOfCandidate.pack(padx=10,pady=10)
274
275     tkinter.Label(numberOfCandidate,textvariable=self.progressStatus["number
276     OfCandidate"]).pack()
277
278     numberOfVote = tkinter.LabelFrame(infoPanel,text="number
279     of vote")
280     numberOfVote.pack(padx=10,pady=10)
281
282     tkinter.Label(numberOfVote,textvariable=self.progressStatus["numberOfVot
283     e"]).pack()
284
285     #-----
286

```

```

273
274     #-----
275     progress = tkinter.LabelFrame(frame,text="Progress")
276     progress.pack(padx=50,pady=10)
277
278     readVote = tkinter.LabelFrame(progress,text="number of
279     read vote")
280     readVote.pack(padx=10,pady=10)
281
282     tkinter.Label(readVote,textvariable=self.progressStatus["readVote"]).pac
283     k()
284
285     currentData = tkinter.LabelFrame(progress,text="current
286     read data")
287     currentData.pack(padx=10,pady=10)
288
289     tkinter.Label(currentData,textvariable=self.progressStatus["currentData"
290     ]).pack()
291
292     #-----
293
294     self.progressBar.pack(fill=tkinter.X)
295     self.progressPanel.mainloop()
296
297     def updateProgress(self,mode:str) -> None:
298
299     self.updateProgressBar(mode)
300
301     if(mode == "countVote"):
302         self.progressStatus["status"].set(mode)
303
304     self.progressStatus["readVote"].set(str(self.counter))
305     currentData = ""
306     for index,vote in enumerate(self.votelist[-1]):
307         if(index == 6):
308             currentData += "..."
309             break
310         currentData += str(vote) + ","
311     self.progressStatus["currentData"].set(currentData)
312     return
313
314     if(mode == "initCandidate"):
315         self.progressStatus["status"].set(mode)
316         return

```

```

310
311         if(mode == "init"):
312             self.progressBarState["status"].set(mode)
313
314 self.progressBarState["numberOfVote"].set(str(self.numberOfVote))
315
316 self.progressBarState["numberOfCandidate"].set(str(self.numberOfCandidate)
317 )
318         return
319
320     def updateProgressBar(self,mode):
321         if(mode == "countVote"):
322             progress = self.progressBarState["initStep"] +
323 self.progressBarState["countVoteStepLength"] * float(self.counter) /
324 self.numberOfVote
325             self.progressBarState["step"].set(int(progress))
326
327 self.progressBarState["prevStep"].set(self.progressBarState["step"].get(
328 ))
329         return
330
331     if(mode == "initCandidate"):
332         progress = self.progressBarState["initStep"] +
333 self.progressBarState["prevStep"].get() +
334 (self.progressBarState["countVoteStepLength"] *
335 float(self.candidateIndex) / self.numberOfCandidate)
336         self.progressBarState["step"].set(int(progress))
337         return
338
339     if(mode == "init"):
340         self.progressBarState["step"].set(int(self.progressBarState["initStep"]))
341     )
342
343         return
344
345     def getData(self,blob:TextIOWrapper) -> None:
346         reader = csv.reader(blob,delimiter=" ")
347
348         self.counter = 0
349         for index,row in enumerate(reader):
350             if(index == 0):

```

```

341         self.numberOfCandidate = int(row[2])
342         self.updateProgress("init")
343         continue
344
345         if(index == 1):
346             self.numberOfVote = int(row[2])
347             self.updateProgress("init")
348             continue
349
350         if(index == 2):
351             # do nothing
352             continue
353
354         new = []
355         for i in range(self.numberOfCandidate):
356             new.append(int(row[i]))
357         self.voteList.append(new)
358
359         if(self.counter == self.numberOfVote):
360             break
361
362         self.counter += 1
363         self.updateProgress("countVote")
364
365
366         self.candidateIndex = 0
367         self.updateProgress("initCandidate")
368         for i in range(self.numberOfCandidate):
369             self.candidateIndex = i
370             self.updateProgress("initCandidate")
371             new = {"id":i + 1,"count":0,"exclude":False}
372             self.candidateList.append(new)
373
374         self.progressBarState["step"].set(int(self.progressBarState["initStep"]))
375
376     def resetCandidateListCount(self) -> None:
377         for aCandidate in self.candidateList:
378             aCandidate["count"] = 0
379
380     def findAllCandidateExcluded(self) -> bool:
381         for aCandidateInfo in self.candidateList:
382             if(not aCandidateInfo["exclude"]): False
383         return True
384

```

```

385     def showInfo(self) -> None:
386         print(self.voteList)
387         print(self.candidateList)
388         print(self.numberOfVote)
389         print(self.numberOfCandidate)
390 from data import data
391 from multiprocessing import Pool
392 import os
393 import tkinter
394 from tkinter import ttk
395 import threading
396 from statusPanel import progressPanel
397
398 def findCandidate(data:data,id:int) -> dict:
399     for ACandidateInfo in data.candidateList:
400         if(ACandidateInfo["id"] == id):
401             return ACandidateInfo
402
403     return None
404
405 def findNoneExculde(data:data,vote) -> int:
406     for aCandidate in vote:
407         info = findCandidate(data,aCandidate)
408         if(not info["exclude"]): return aCandidate
409     return None
410
411 def countCaandidate(data:data,voteList) -> None:
412     for aCandidateInfo in data.candidateList:
413         for aCandidate in voteList:
414             if(aCandidate == aCandidateInfo["id"]):
415                 aCandidateInfo["count"] += 1
416
417 def findWinner(data:data) -> dict:
418     threshold = int(data.numberOfVote / 2) + 1
419     for aCandidateInfo in data.candidateList:
420         if(aCandidateInfo["count"] >= threshold and not
aCandidateInfo["exclude"]):
421             return aCandidateInfo
422     return None
423
424 def findExclude(data:data) -> None:
425     minVote = data.candidateList[0]
426

```

```

427     counter = 1
428     while(minVote["exclude"]):
429         minVote = data.candidateList[counter]
430         counter += 1
431
432     for aCandidateInfo in data.candidateList[counter:]:
433         if(minVote["count"] > aCandidateInfo["count"] and not
aCandidateInfo["exclude"]):
434             minVote = aCandidateInfo
435
436     minVote["exclude"] = True
437
438
439
440 def process(data:data,statusPanel:progressPanel) -> str:
441
442     winner = None
443
444     counter = 0
445     while(1):
446         voteList = []
447
448         for vote in data.voteList:
449             voteList.append(findNoneExculde(data,vote))
450         print("find non excluded candidate")
451         statusPanel.updateProgress("findNonExclude")
452
453         # slow
454         countCaandidate(data,voteList)
455         print("count candidate")
456         statusPanel.updateProgress("countCandidate")
457
458         winner = findWinner(data)
459         print("found winner")
460         statusPanel.updateProgress("foundWinner")
461
462         if(winner != None): break
463         findExclude(data)
464         print("find to exclude")
465         statusPanel.updateProgress("findToExclude")
466
467         data.resetCandidateListCount()
468         print("reset vote counter of candidates")
469         statusPanel.updateProgress("resetVote")
470

```

```

471         if(not data.findAllCandidateExcluded()):
472             return None
473         print("check all candidates are excluded")
474         statusPanel.updateProgress("cheCAllCandidate")
475
476         print("progress: " + str(counter) + "\n\n")
477         counter += 1
478
479     return winner
480
481 import tkinter
482 from data import *
483 import threading
484 from tkinter import messagebox
485
486
487
488 class errCheckProgressPanel:
489     def __init__(self) -> None:
490         self.ProgressInit()
491
492     def ProgressInit(self) -> None :
493         self.root = tkinter.Tk()
494         self.root.title("error check status")
495         self.root.geometry("400x200")
496
497         self.progressStatus = {
498             "status": tkinter.StringVar(self.root,"please
wait..."),
499             "progress": tkinter.IntVar(self.root,0),
500             "checkVoteData":tkinter.IntVar(self.root,0)
501         }
502
503         self.progressBar =
504         ttk.Progressbar(self.root,variable=self.progressStatus["progress"],maxim
um=100)
505
506         self.checkVoteDataBar =
507         ttk.Progressbar(self.root,variable=self.progressStatus["checkVoteData"],
maximum=100)
508
509         self.checkVoteDataLabel =
510         tkinter.Label(self.root,text="vote row check progress")
511
512     def ProgressShow(self) -> None :
```

```

510         status = tkinter.LabelFrame(self.root,text="status")
511         status.pack(padx=10,pady=10)
512
513         tkinter.Label(status,textvariable=self.progressStatus["status"]).pack()
514
515         self.progressBar.pack(fill=tkinter.X)
516         threading.Thread(self.root.mainloop()).start()
517
518     def ProgressHide(self) -> None :
519         self.root.destroy()
520
521     def showCheckVoteDataInfo(self) -> None:
522         self.checkVoteDataLabel.pack()
523         self.checkVoteDataBar.pack(fill=tkinter.X)
524
525     def hideCheckVoteDataInfo(self) -> None:
526         self.checkVoteDataLabel.forget()
527         self.checkVoteDataBar.forget()
528
529     def showErrorMessage(self):
530         text = "err occured in " +
531         self.progressStatus["status"].get() +" err check process\n"
532         text += "Please check your data file is correct."
533         messagebox.askokcancel(title="error",message=text)
534
535     def ProgressUpdate(self,mode:str,data:dict) -> None :
536         if(mode == "wordCheck"):
537             self.progressStatus["status"].set(mode)
538
539         self.progressStatus["progress"].set(data["progress"])
540
541         # -----
542         if(mode == "valueCheck"):
543             if(data["status"] == "start"):
544                 self.progressStatus["status"].set(mode)
545
546         self.progressStatus["progress"].set(data["progress"])
547
548         if(data["status"] == "onProcess"):
549             index = data["index"]
550             maxVal = data["max"]
551             progress = int(100 * (index / float(maxVal)))
552             self.progressStatus["progress"].set(progress)
```

```

550         if(data["status"] == "end"):
551
self.progressStatus["progress"].set(data["progress"])
552         # -----
553
554         if(mode == "checkVoteData"):
555             if(data["status"] == "start"):
556                 self.progressStatus["status"].set(mode)
557
self.progressStatus["progress"].set(data["progress"])
558                 self.showCheckVoteDataInfo()
559
560             if(data["status"] == "onProcessEntire"):
561                 self.progressStatus["status"].set(mode + " " +
str(data["counter"]) + "/" + str(data["numberOfVote"]))
562                 counter = data["counter"]
563                 numberOfVote = data["numberOfVote"]
564                 progress = int(100 * ((counter - 2) /
float(numberOfVote)))
565                 self.progressStatus["progress"].set(progress)
566
567             if(data["status"] == "onProcessVoteRow"):
568                 index = data["index"]
569                 maxVal = data["numberOfCandidate"]
570                 progress = int(100 * (index / float(maxVal)))
571
self.progressStatus["checkVoteData"].set(progress)
572
573             if(data["status"] == "end"):
574                 self.progressStatus["status"].set(mode)
575
self.progressStatus["progress"].set(data["progress"])
576                 self.hideCheckVoteDataInfo()
577
578
579     class progressPanel:
580
581         def __init__(self,root:tkinter.Tk) -> None:
582             self.root = root
583             self.initProgress()
584
585         def initProgress(self) -> None:
586             self.processProgressPanel =
tkinter.LabelFrame(self.root,text="process progress status")

```

```

587
588         self.processStatus = {
589             "status":
tkinter.StringVar(self.processProgressPanel,"None"),
590             "progress":
tkinter.IntVar(self.processProgressPanel,0),
591             "findNonExclude": 10,
592             "countCandidate": 20,
593             "foundWinner": 30,
594             "findToExclude": 40,
595             "resetVote": 50,
596             "checAllCandidate": 60,
597             "total": 60
598         }
599
600         self.progressBar =
ttk.Progressbar(self.processProgressPanel,variable=self.processStatus["p
rogess"],maximum=self.processStatus["total"])
601
602         #-----
603         frameRow1 = tkinter.Frame(self.processProgressPanel)
604         frameRow1.pack(padx=10,pady=10)
605
606         self.status =
tkinter.LabelFrame(frameRow1,text="status")
607         self.status.pack()
608
tkinter.Label(self.status,textvariable=self.processStatus["status"]).pac
k()
609         #-----
610         self.progressBar.pack(fill=tkinter.X)
611
612
613         def updateProgress(self,mode:str) -> None:
614
self.progressStatus["progress"].set(self.processStatus[mode])
615             self.processStatus["status"].set(mode)
616
617         def showPanel(self) -> None:
618
self.processProgressPanel.pack(fill=tkinter.X,padx=10,pady=10)
619
620
621
622

```



```

623
624
625
626
627
628
629
630
631
632
633
634
635
636
637 def
updatePanel(state:tkinter.LabelFrame,refList:list,datas:data) -> None:
638     interval = int(1.0 / 5 * 1000)
639     for ref,AcandidateInfo in zip(refList,datas.candidateList):
640         # code of first line for the debug purpose
641         ref["id"].set("id: " + "%4d" % (AcandidateInfo["id"]))
642         ref["count"].set("count: " + "%4d" %
(AcandidateInfo["count"]))
643         ref["exclude"].set("exclude: " +
str(AcandidateInfo["exclude"]))
644
state.after(interval,lambda :updatePanel(state,refList,datas))
645
646
647 def candidateStatePanel(root:tkinter.Tk,datas:data) -> None:
648     mainFrame = tkinter.Frame(root)
649     mainFrame.pack()
650
651     scrollContainer = tkinter.Canvas(mainFrame)
652     scrollContainer.grid(padx=20,pady=20,row=0,column=0)
653
654     scroll =
tkinter.Scrollbar(mainFrame,command=scrollContainer.yview)
655     scroll.grid(row=0,column=1,sticky=tkinter.NS)
656     scrollContainer.configure(yscrollcommand=scroll.set)
657
658     state = tkinter.LabelFrame(scrollContainer,text="candidate
state")
659     state.pack(padx=20,pady=20)
660
661     refList = []
662     # add candidate
663     for AcandidateInfo in datas.candidateList:

```

```

664         ref = {}
665         frame = tkinter.Frame(state)
666         frame.pack()
667
668         candidateID = tkinter.StringVar(state,"id: " + "%4d" %
(AcandidateInfo["id"]))
669         count = tkinter.StringVar(state,"count: " + "%4d"
% (AcandidateInfo["count"]))
670         exclude = tkinter.StringVar(state,"exclude: " +
str(AcandidateInfo["exclude"]))
671
672         ref["id"] = (candidateID)
673         ref["count"] = (count)
674         ref["exclude"] = (exclude)
675
676         tkinter.Label(frame,textvariable=candidateID).pack(side=tkinter.LEFT)
677         tkinter.Label(frame,textvariable=count).pack(side=tkinter.LEFT)
678         tkinter.Label(frame,textvariable=exclude).pack()
679
680         refList.append(ref)
681
682         state.update_idletasks()
683         root.update_idletasks()
684         scrollContainer.create_window(0,0,window=state,anchor="nw")
685
scrollContainer.config(scrollregion=(0,0,state.winfo_reqwidth(),state.wi
nfo_reqheight()))
686         if(state.winfo_reqheight() > 200):
687
scrollContainer.configure(width=state.winfo_reqwidth(),height=200)
688         else:
689
scrollContainer.configure(width=state.winfo_reqwidth(),height=state.winf
o_reqheight())
690
691         updater =
threading.Thread(target=updatePanel,args=(state,refList,datas))
692         updater.start()

```