


電子工学科学生実験における選択テーマ振り分けプログラムの作成

実験教育支援センター
門出 康孝

背景

■ 電気電子工学実験第2は選択制

◆ 定員を考慮しつつ希望順に学生を配置するのは非常に手間

◆ 名簿の情報から自動で振り分ける方法があると便利

出席番号	氏　名	ヨミガナ	10/1	10/8	10/15	10/22	10/29	11/5	11/26	12/3	12/10	12/17	12/24	1/7
			映像化システム			半導体デバイス			ロボットビジョンプログラミング					
			映像化システム			半導体デバイス			ロボットビジョンプログラミング					
			レーザ			半導体デバイス			ロボットビジョンプログラミング					
			多脚ロボット制御						半導体デバイス			映像化システム		
			映像化システム			PCM通信			ロボットビジョンプログラミング					
			アンテナ製作			半導体デバイス			映像化システム			PCM通信		
			半導体デバイス			レーザ			PCM通信			映像化システム		
			半導体デバイス			PCM通信			ロボットビジョンプログラミング					
			映像化システム			レーザ			ロボットビジョンプログラミング					
			アンテナ製作			半導体デバイス			PCM通信			レーザ		
			半導体デバイス			PCM通信			レーザ			映像化システム		
			PCM通信			映像化システム			ロボットビジョンプログラミング					
			半導体デバイス			映像化システム			ロボットビジョンプログラミング					
			映像化システム			レーザ			ロボットビジョンプログラミング					
			多脚ロボット制御						映像化システム			レーザ		

目的

- 選択実験自動振り分けプログラムの作成
  - 作成者以外でも使いやすいシンプルなインターフェース
  - 編集しやすい結果の出力
  - 動作環境に依存しない配布可能な形式
- 自身のプログラミング能力向上
  - 使用経験のないツールを使用し、プログラム作成能力の幅を広げる

仕様の決定

- 使用言語:Python
  - 数学系のライブラリが充実している
  - 文法がシンプルで使いやすい

- 結果出力形式:CSVファイル
  - Excel等で編集しやすく、表の掲示に向いている

- 拡張子:EXEファイル
  - Pythonのままだと配布時に環境構築をする必要がある
  - Windowsで実行可能なファイル

電気電子工学実験第2概要

- 選択テーマ3つとコアテーマ4つの中から3～4テーマを選択
  - 選択テーマは1つのみ選択可能
  - コアテーマのみの選択も可能

- 選択テーマの多脚ロボットとロボットビジョンプログラミングは2タームに渡って実施

選択テーマ

多脚ロボット

ロボットビジョン
プログラミング

アンテナ製作

コアテーマ

半導体デバイス

レーザ

PCM通信

映像化システム

PCM : Pulse Code Modulation

基本方針

■ 選択テーマを割当

多脚ロボット制御			
		ロボットビジョンプログラミング	
	アンテナ製作		

■ コアテーマの第1希望を空いているタームにランダム割当

◆ 各タームの定員を超えてしまう場合は抽選

多脚ロボット制御		半導体デバイス	
	PCM通信	ロボットビジョンプログラミング	
	アンテナ製作		PCM通信
		PCM通信	

■ 第2希望以降も同様に割当

多脚ロボット制御		半導体デバイス	映像化システム
映像化システム	PCM通信	ロボットビジョンプログラミング	
半導体デバイス	アンテナ製作	映像化システム	PCM通信
半導体デバイス	レーザ	PCM通信	映像化システム

プログラムへの名簿入力形式

■ CSVファイル

◆ 選択テーマは希望する実験の番号を、コアテーマは希望順位を入力

◆ 1行目はインデックスとして読み飛ばす

選択	半導体	レーザ	PCM	映像化
2	1	4	3	2
2	2	4	3	1
2	1	2	4	3
1	1	3	4	2
2	3	4	2	1
3	1	4	2	3
4				
2	1	3	2	4
2	3	2	4	1
3	1	2	3	4

選択テーマ
1:多脚ロボット制御
2:ロボットビジョンプログラミング
3:アンテナ製作
4:コアテーマのみ

コアテーマ
第1~第4希望までの入力
選択実験を選ばなかった学生は
全てのコアテーマを実施するため
希望順位は空欄

選択テーマ割当

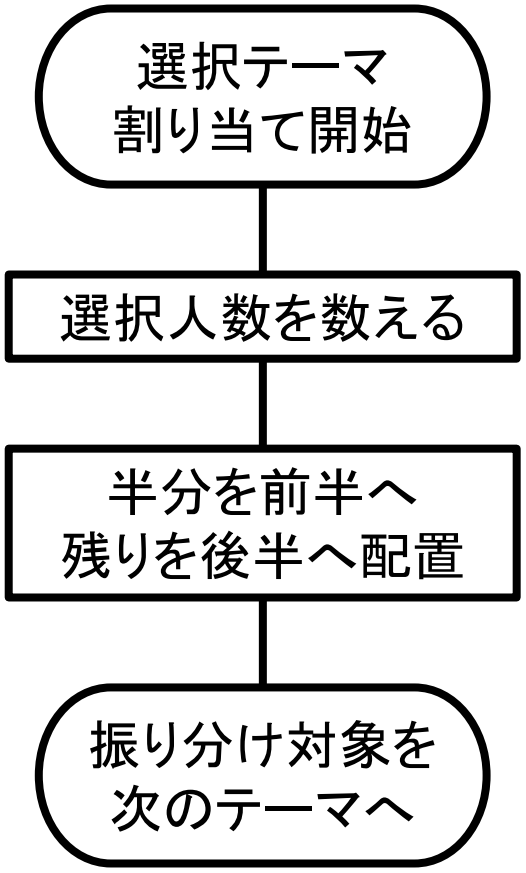
- 選択テーマを選んだ人数の半分を1,2タームに
  残り半分を3,4タームに配置
  - アンテナ実験は1タームでの実験なので、選択人数の1/4ずつ配置

選択	半導体	レーザ	PCM	映像化
2	1	4	3	2
2	2	4	3	1
2	1	2	4	3
1	1	3	4	2
2	3	4	2	1
3	1	4	2	3
4				
2	1	3	2	4
2	3	2	4	1
3	1	2	3	4

第1ターム	第2ターム	第3ターム	第4ターム
ロボットビジョン			
ロボットビジョン			
ロボットビジョン			
多脚ロボット制御			
		ロボットビジョン	
アンテナ			
		ロボットビジョン	
		ロボットビジョン	
		アンテナ	

選択テーマ割当

- 選択テーマを選んだ人数の半分を1,2タームに
  残り半分を3,4タームに配置
  - アンテナ実験は1タームでの実験なので、選択人数の1/4ずつ配置




```
# 選択実験の割り当て 選択人数の半分は前半残りを後半

for i in range(nin):
  if data[i][0] == '1':
    if takyaku_count <= (takyaku_num/2):
      arr[i][0] = 'takyaku'
      arr[i][1] = 'takyaku'
      takyaku_count +=1
    else:
      arr[i][2] = 'takyaku'
      arr[i][3] = 'takyaku'
```


コアテーマ第1希望割当

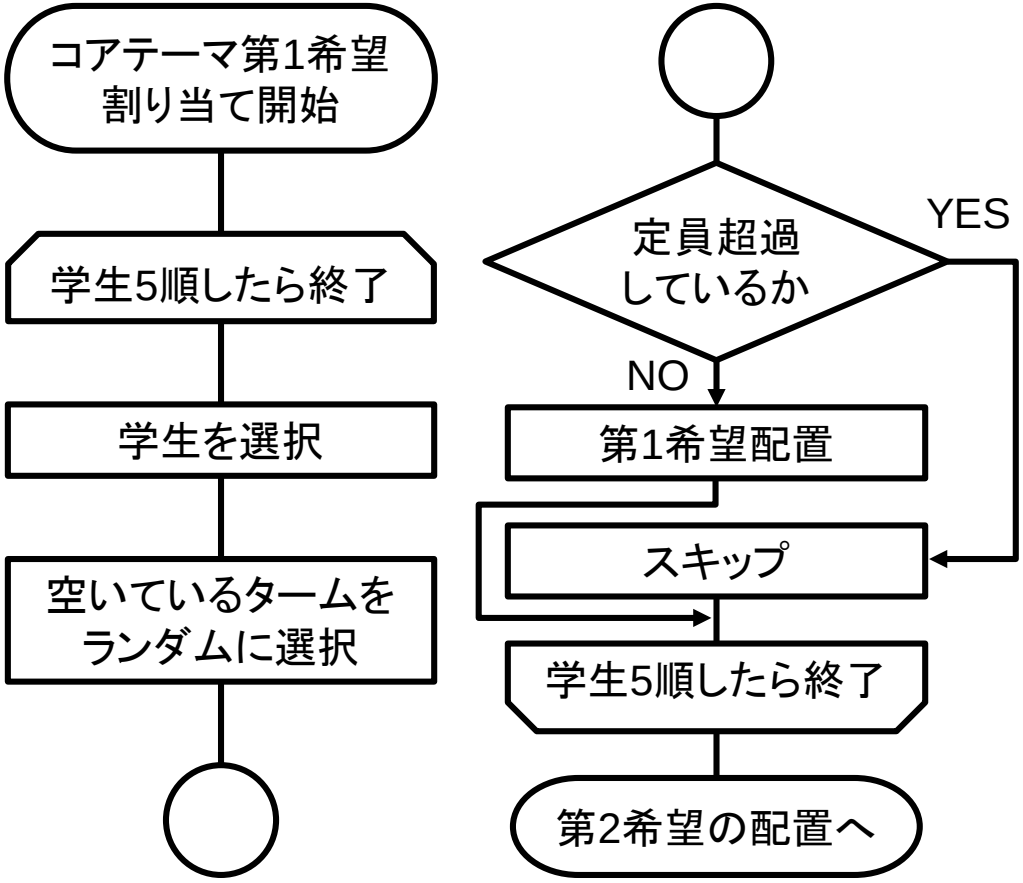
■ 学生を選択し第1希望の実験を空いているタームに配置
- ◆ 配置の際、各タームの定員を超えている場合はスキップ
- ◆ コアテーマのみの学生はワイルドカードとして最後の調整に使用

選択	半導体	レーザ	PCM	映像化
2	1	4	3	2
2	2	4	3	1
2	1	2	4	3
1	1	3	4	2
2	3	4	2	1
3	1	4	2	3
4				
2	1	3	2	4
2	3	2	4	1
3	1	2	3	4

第1ターム	第2ターム	第3ターム	第4ターム
ロボットビジョン			半導体
ロボットビジョン		映像化	
ロボットビジョン			半導体
多脚ロボット制御		半導体	
映像化		ロボットビジョン	
アンテナ	半導体		
半導体		ロボットビジョン	
	映像化	ロボットビジョン	
	半導体	アンテナ	

コアテーマ第1希望割当

■ 学生を選択し第1希望の実験を空いているタームに配置
- ◆ 配置の際、各タームの定員を超えている場合はスキップ
- ◆ コアテーマのみの学生はワイルドカードとして最後の調整に使用




```python
def function1(kibou = 1):
  # 第1希望割り当て関数を5回繰り返す
  for d in range(5):
    for index in range(4):
      num = 0

      # 各テーマの第1希望者数をカウント
      for i in range(removed_arr_len):
        if c[i][index + 1] == str(kibou):
          num += 1
      # 希望者数が(定員-選択実験希望なし人数)以下の場合、重複がないことを確認し空いているタームに第1希望
      if num <= teiin[index]*4 - count_null:
        for i in range(removed_arr_len):
          if theme[index] in removed_arr[i]:
            pass
          elif c[i][index + 1] == str(kibou):
            random.shuffle(list)
            for e in list:
              if removed_arr[i][e] == 0 and count[index][e] < teiin[index] - null_div:
                removed_arr[i][e] = theme[index]
                count[index][e] += 1
                break
```


コアテーマ第2～第4希望割当

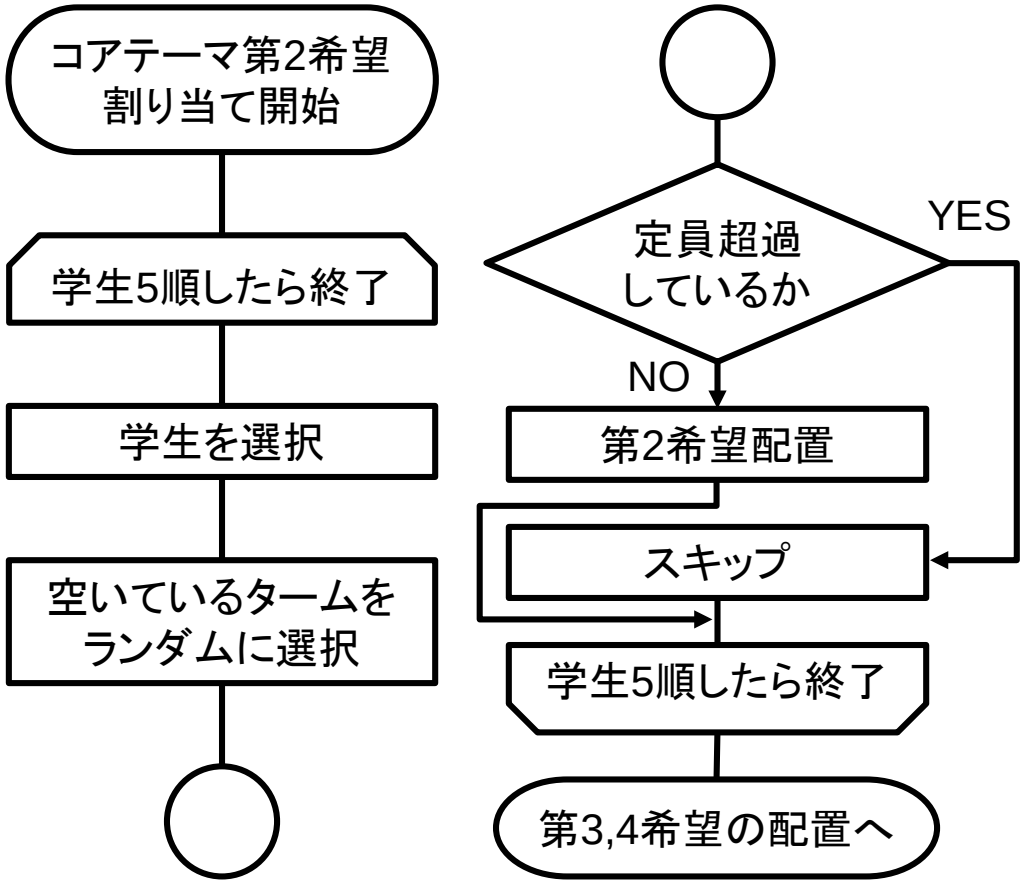
- 第1希望と同様に学生を選択し、第2希望の実験を空いているタームに配置
  - 第3、第4希望の実験も同様に配置

選択	半導体	レーザ	PCM	映像化
2	1	4	3	2
2	2	4	3	1
2	1	2	4	3
1	1	3	4	2
2	3	4	2	1
3	1	4	2	3
4				
2	1	3	2	4
2	3	2	4	1
3	1	2	3	4

第1ターム	第2ターム	第3ターム	第4ターム
ロボットビジョン		映像化	半導体
ロボットビジョン		映像化	半導体
ロボットビジョン		レーザ	半導体
多脚ロボット制御		半導体	映像化
映像化	PCM通信	ロボットビジョン	
アンテナ	半導体	映像化	PCM通信
半導体	PCM通信	ロボットビジョン	
レーザ	映像化	ロボットビジョン	
PCM通信	半導体	アンテナ	レーザ

コアテーマ第2～第4希望割当

- 第1希望と同様に学生を選択し、第2希望の実験を空いているタームに配置
  - 第3、第4希望の実験も同様に配置




```python
def function234(kibou):
  # 割り当て関数を5回繰り返す
  for d in range(5):
    for index in range(4):
      num = 0
      # すでに(定員-選択実験希望なし人数)を上回っている場合、この先を棄却
      if sum(count[index]) < teiin[index]*4 - count_null:
        # 希望順位の人数をカウント
        for i in range(removed_arr_len):
          if c[i][index + 1] == str(kibou):
            num += 1
        # これまでに挿入された人数と希望人数の和が定員に満たない場合、重複がないことと(タームごとの定員-希望なし
        if sum(count[index]) + num <= teiin[index]*4:
          for i in range(removed_arr_len):
            if theme[index] in removed_arr[i]:
              pass
            elif c[i][index + 1] == str(kibou):
              random.shuffle(list)
              for e in list:
                if removed_arr[i][e] == 0 and count[index][e] < teiin[index] - null_div:
                  removed_arr[i][e] = theme[index]
                  count[index][e] += 1
                  break
```

コアテーマ希望割当の評価

■ 第4希望配置終了後、評価を実施

◆ ランダム配置の結果各タームに偏りが生じ、定員オーバーで空欄ができている場合は再度第1希望からコアテーマ割当を実施

◆ 問題なく割当できていたら次のステップに移行

第1ターム	第2ターム	第3ターム	第4ターム
ロボットビジョン		映像化	半導体
ロボットビジョン		映像化	半導体
ロボットビジョン		レーザ	半導体
多脚ロボット制御		半導体	映像化
映像化	PCM通信	ロボットビジョン	
アンテナ	半導体	映像化	PCM通信
半導体	PCM通信	ロボットビジョン	
レーザ	映像化	ロボットビジョン	
PCM通信	半導体	アンテナ	レーザ

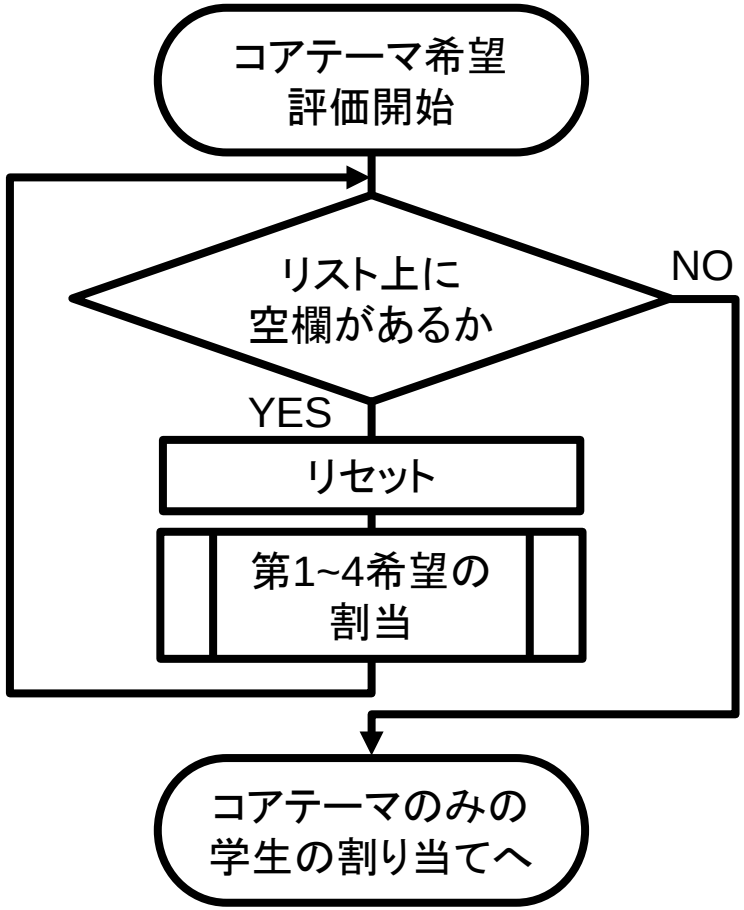
次のステップへ

第1ターム	第2ターム	第3ターム	第4ターム
ロボットビジョン		映像化	半導体
ロボットビジョン		映像化	
ロボットビジョン		レーザ	半導体
多脚ロボット制御		半導体	映像化
映像化	PCM通信	ロボットビジョン	
アンテナ	半導体		PCM通信
半導体	PCM通信	ロボットビジョン	
レーザ	映像化	ロボットビジョン	
PCM通信	半導体	アンテナ	レーザ

再度割当

コアテーマ希望割当の評価

- 第4希望配置終了後、評価を実施
  - ランダム配置の結果各タームに偏りが生じ、定員オーバーで空欄ができている場合は再度第1希望からコアテーマ割当を実施
  - 問題なく割当できていたら次のステップに移行




```
while 0 in flatten:
  # カウントリセット
  count = [[0 for i in range(term)] for j in range(4)]
  # 行列リセット
  removed_arr = copy.deepcopy(d_arr)
  # 第1希望~第4希望割り当て関数の実施
  function1(1)
  function234(2)
  function234(3)
  function234(4)
  # 現在の割り当て人数の表示
  print count
  # 行列をベクトル化し、while文の評価へ
  flatten = [flat for inner in removed_arr for flat in inner]
```


コアテーマのみの学生を割当

■ 選択実験希望者の割当終了後コアテーマのみの学生を割当

◆ ランダムでテーマを選び、各タームの定員を超えていないことを確認してから割当を実施

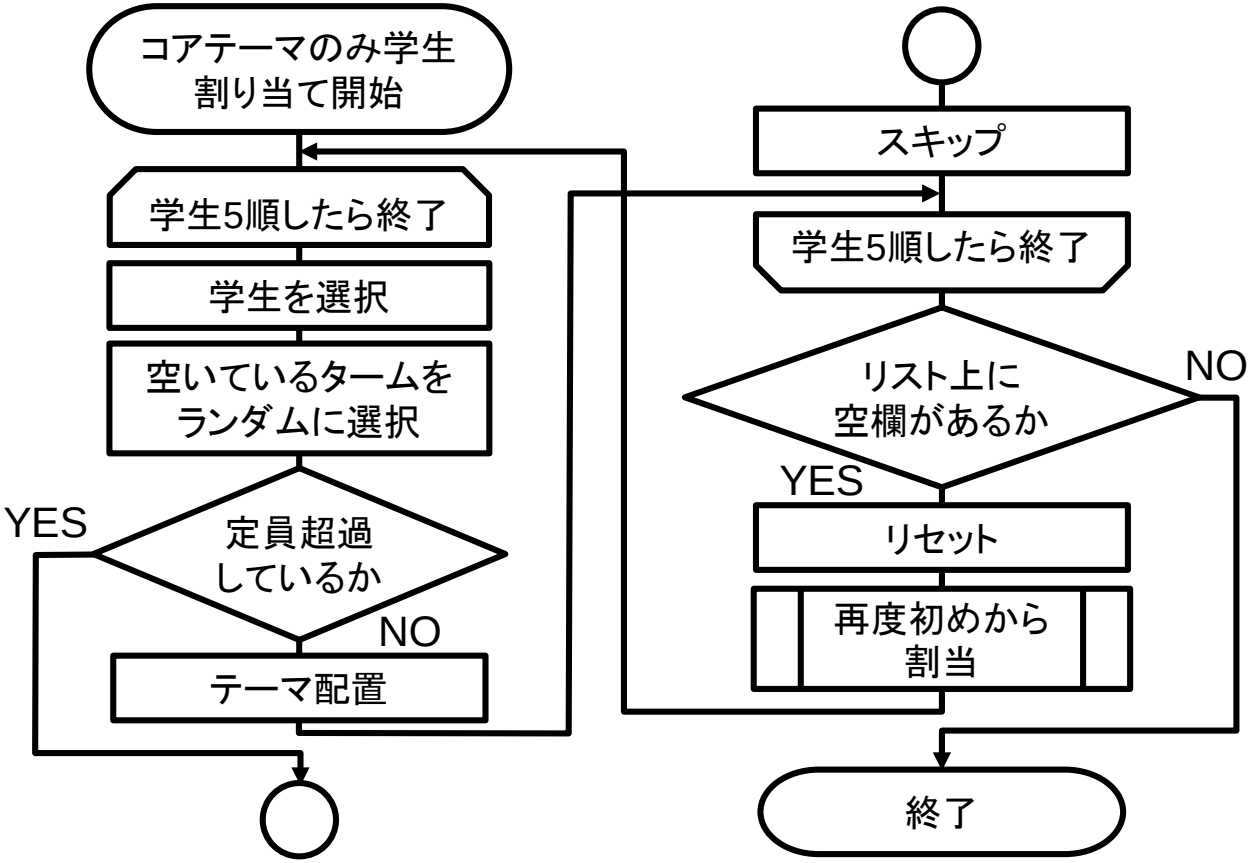
◆ 一定回数繰り返した結果、定員超過のためすべての学生に割当が行えない場合については再度コアテーマ第１希望割当から実施

選択	半導体	レーザ	PCM	映像化
2	1	4	3	2
2	2	4	3	1
2	1	2	4	3
1	1	3	4	2
2	3	4	2	1
3	1	4	2	3
4				
2	1	3	2	4
2	3	2	4	1
3	1	2	3	4

第1ターム	第2ターム	第3ターム	第4ターム
ロボットビジョン		映像化	半導体
ロボットビジョン		映像化	半導体
ロボットビジョン		レーザ	半導体
多脚ロボット制御		半導体	映像化
映像化	PCM通信	ロボットビジョン	
アンテナ	半導体	映像化	PCM通信
半導体	レーザ	PCM通信	映像化
半導体	PCM通信	ロボットビジョン	
レーザ	映像化	ロボットビジョン	
PCM通信	半導体	アンテナ	レーザ

コアテーマのみの学生を割当

- 選択実験希望者の割当終了後コアテーマのみの学生を割当
  - ランダムでテーマを選び、各タームの定員を超えていないことを確認してから割当を実施
  - 一定回数繰り返した結果、定員超過のためすべての学生に割当が行えない場合については再度コアテーマ第1希望割当から実施



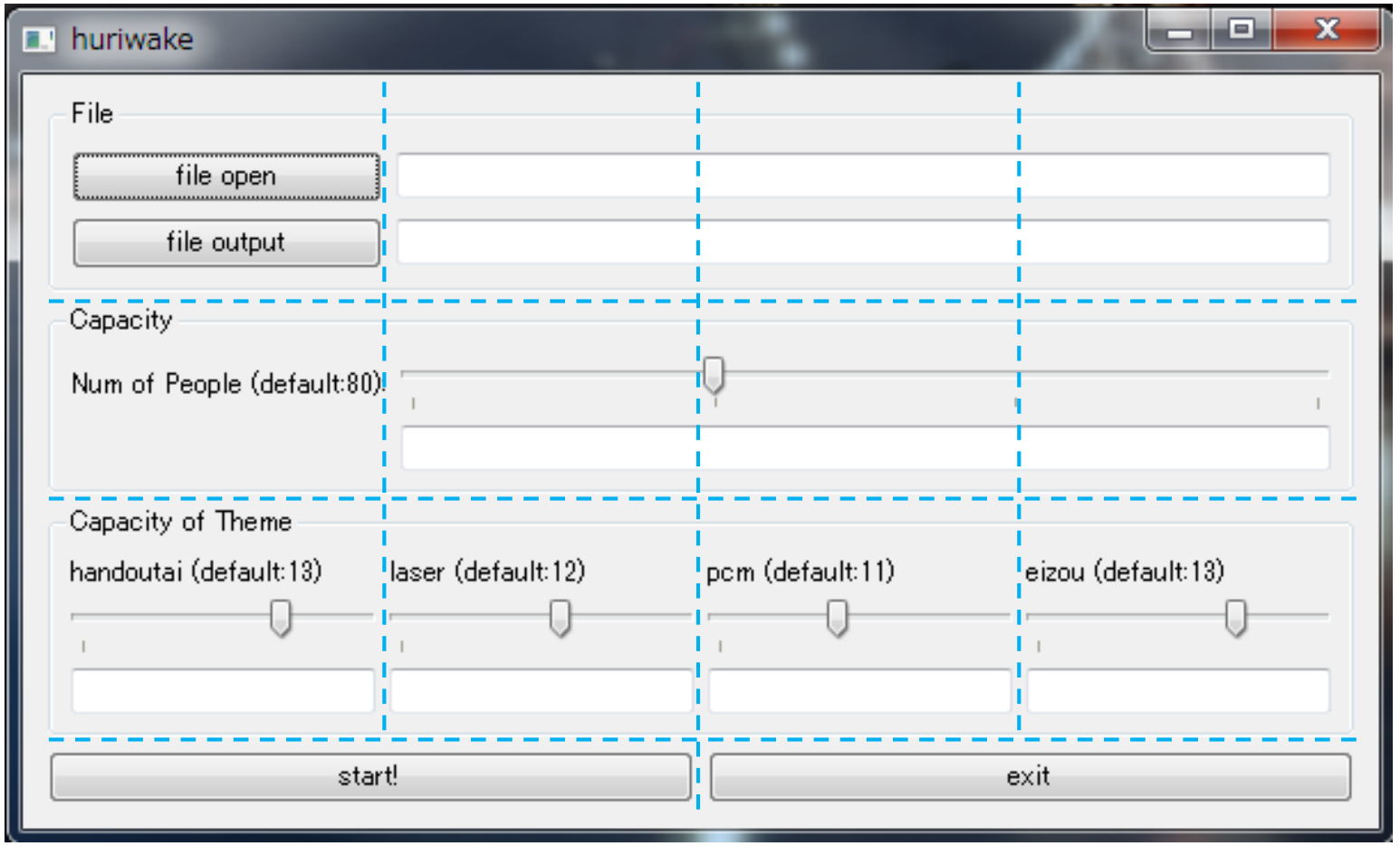

```
# 希望を出していない人に対しランダムにテーマを配置していく 最終出力
while 0 in flatten_arr:
  arr = copy.deepcopy(arr_shuffle)
  count = copy.deepcopy(count_shuffle)

  for d in range(5):
    theme_shuffle = copy.deepcopy(theme)
    random.shuffle(theme_shuffle)

    for i in range(nin):
      for j in range(term):
        index = theme.index(theme_shuffle[j])
        if arr[i][j] == 0:
          if theme_shuffle[j] in arr[i]:
            pass
          elif count[index][j] >= teiin[index] :
            pass
          else:
            arr[i][j] = theme_shuffle[j]
            count[index][j] +=1
```

インターフェース

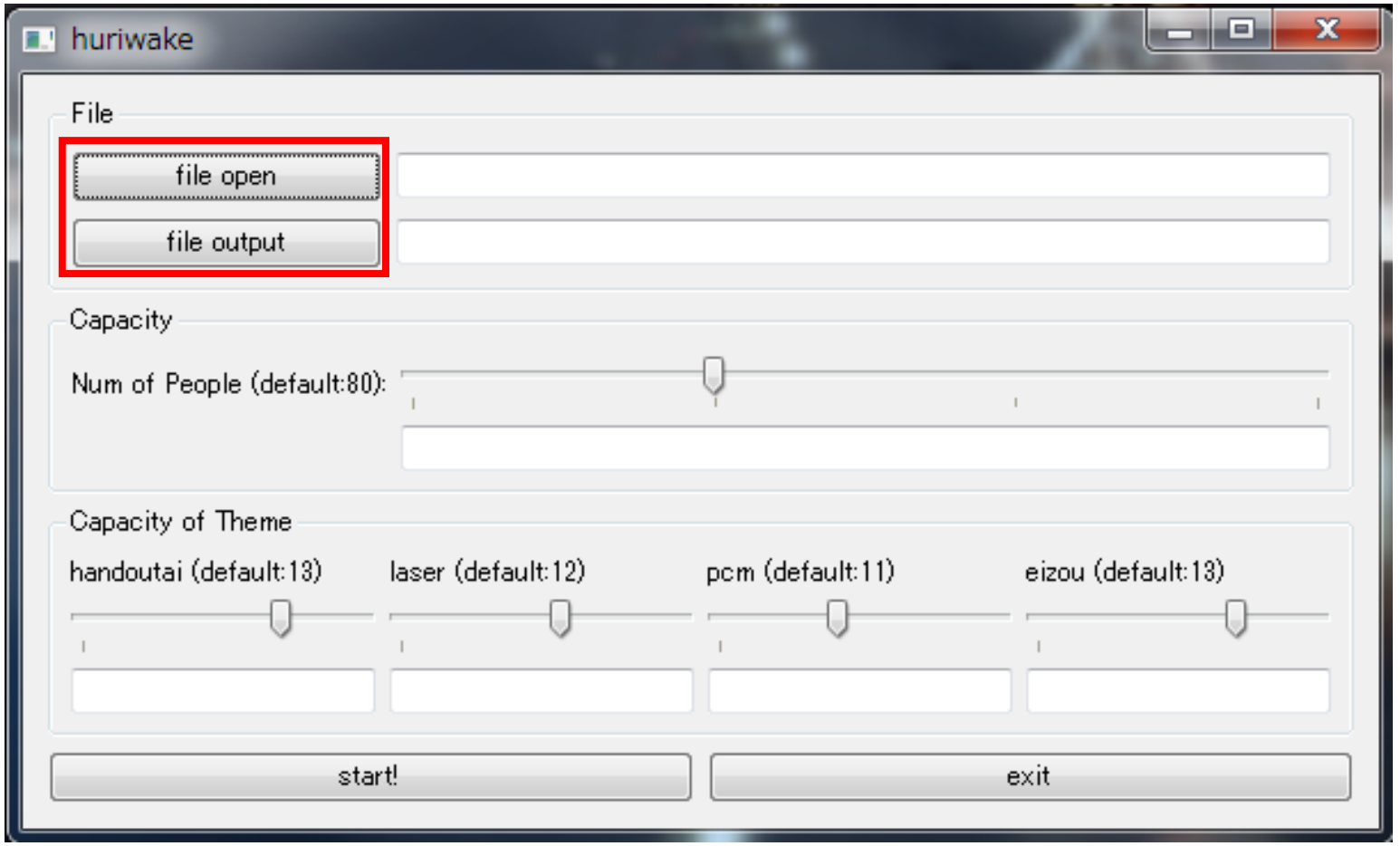
- PyQt4を利用したGUIを採用
  - ボタン・スライダーはマス目状に配置



GUI : Graphical User Interface

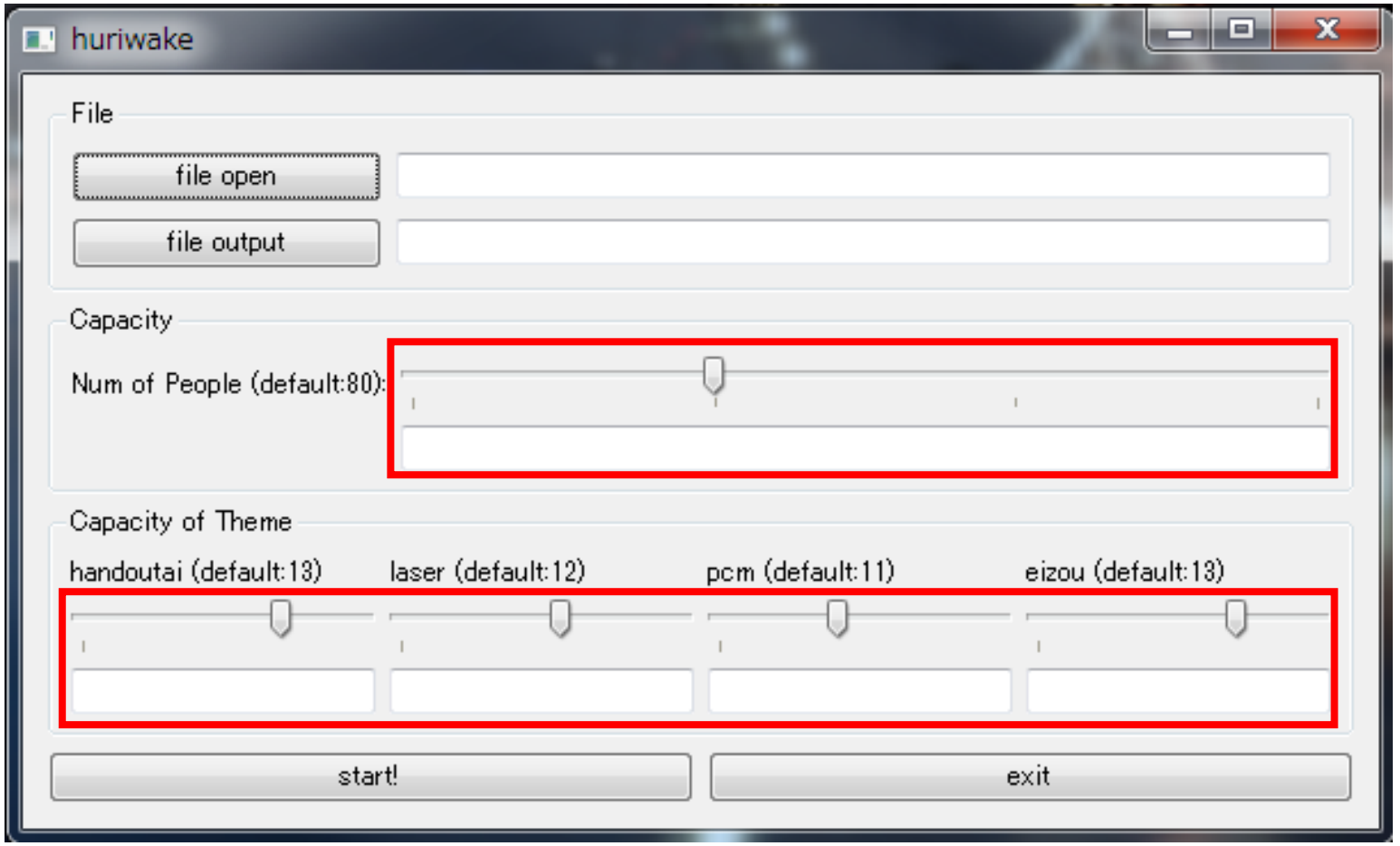
インターフェース

■ ボタンクリックでファイルダイアログが開きファイル名を取得



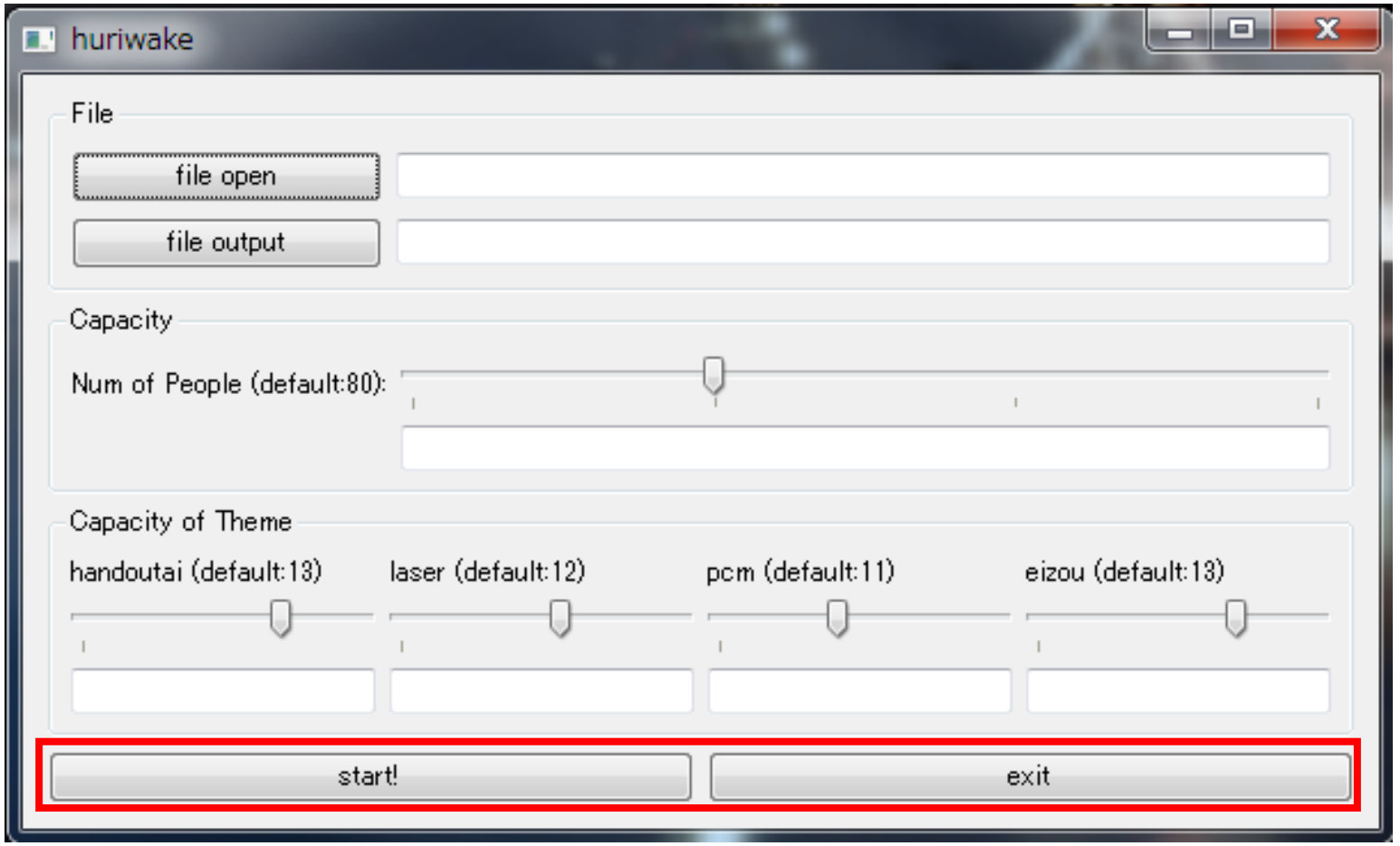
インターフェース

■ 実験履修者数・コアテーマの実験定員はスライダーで指定



インターフェース

- Startボタンで振り分け開始
- Exitボタンでアプリケーション終了



操作方法

結果

■ 下図のようなCSVファイルが生成

◆ 学生氏名・出席番号・色付けなどの編集をして掲示し、完成

robo	robo	eizou	handoutai
robo	robo	pcm	eizou
robo	robo	handoutai	laser
takyaku	takyaku	handoutai	eizou
robo	robo	eizou	pcm
antena	handoutai	pcm	laser
pcm	handoutai	eizou	laser
robo	robo	handoutai	pcm
robo	robo	laser	eizou
antena	handoutai	laser	pcm

比較

■ 振り分けプログラム vs 人

- ◆ 学生80名の振り分けに対しての作業時間を測定
- ◆ 振り分けプログラムは振り分け時間の10回平均を、人は1回の振り分け時間を測定し比較

■ PCスペック

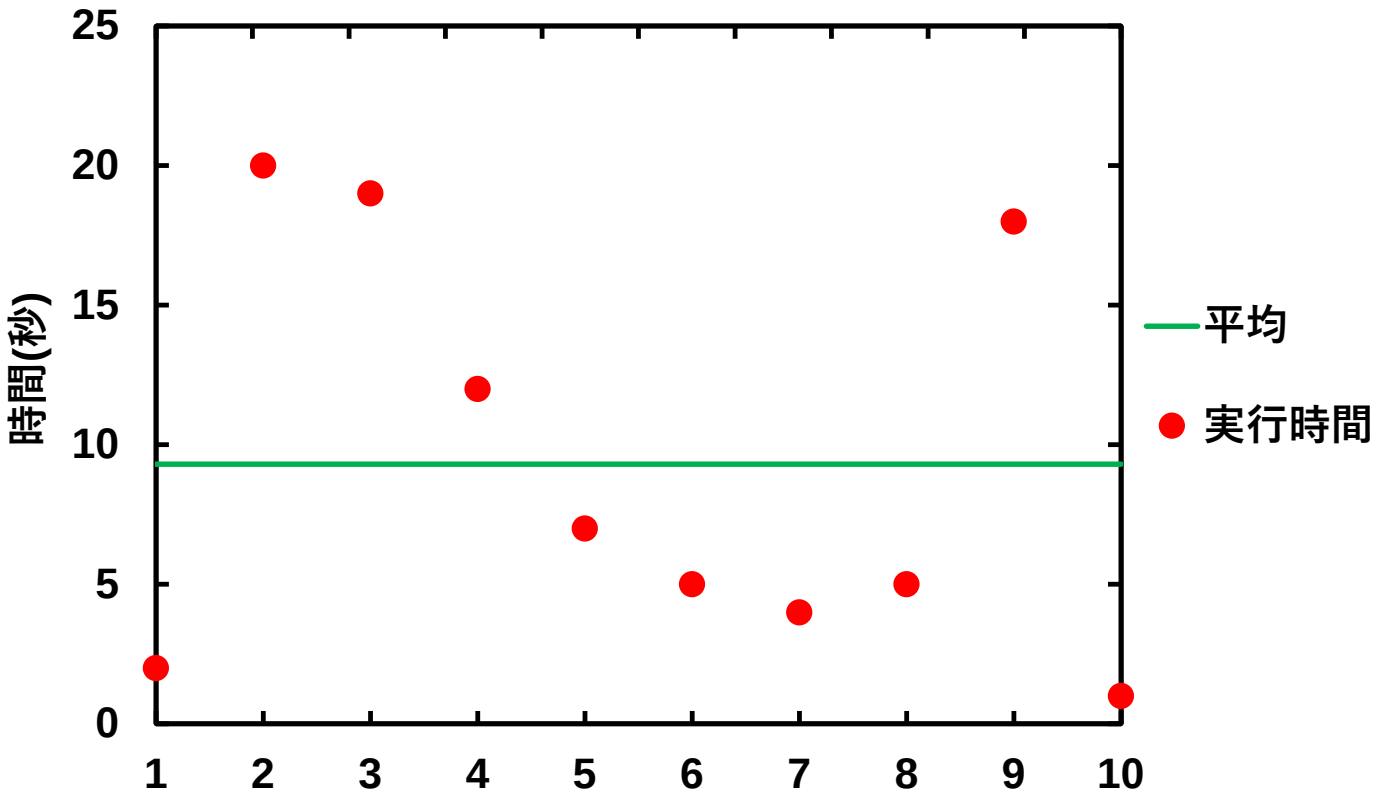
- ◆ OS : Windows 7 Professional 64bit
- ◆ CPU : Intel® Core™ i5-5200U CPU @2.20GHz
- ◆ GPU : Intel® HD Graphics 5500
- ◆ メモリ : 8.00GB

■ 人

- ◆ 電気系共通実験室技術系職員

結果

- 振り分けプログラム
  - 10回平均で測定
  - 平均9.3秒で終了
- 人
  - 1回の試行で測定
  - 56分で終了



プログラム実行時間と平均値

大幅な作業時間削減を達成

まとめ

- 選択実験自動振り分けプログラムを作成した
  - GUIを採用し、直感的に分かりやすいアプリケーションを作成した
  - 配布可能なEXEファイルを作成し、環境構築の手間を減らした
  - 結果はCSV出力とし、表掲示に編集しやすい形にした
  - 選択実験振り分け作業時間を大幅に削減した

- 自身のプログラミング能力を向上させた
  - これまで使用経験の無かったPyQt4などのツールを使い GUIプログラミングができるようになった

課題と今後の展望

- 学生数の増加や希望の偏りがあると解が出づらい
  - ランダムにテーマを配置し、都合が良い結果が出るまで
    やり直しているので、効率が良くない

- 振り分け実行中に動作中であることが分かるようにする
  - Startボタンを押した後、動作中であることが分かるような仕組みを
    取り入れたい