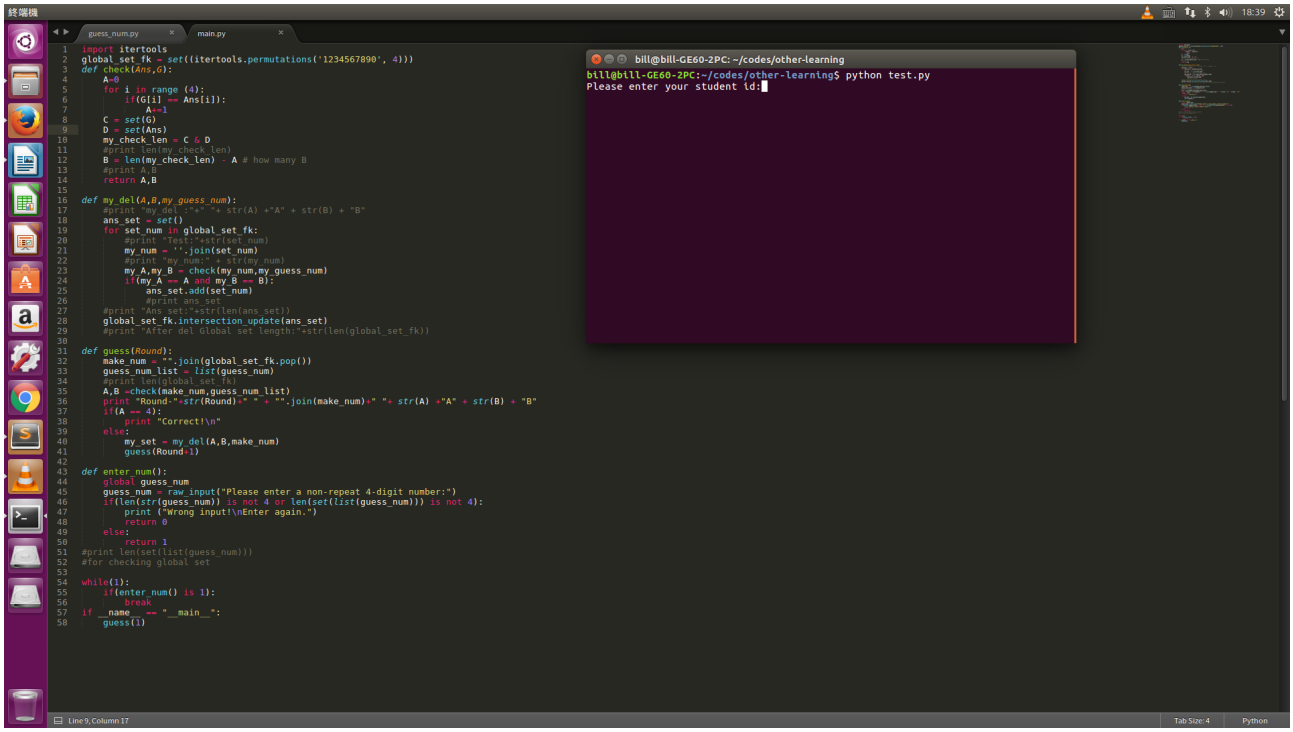
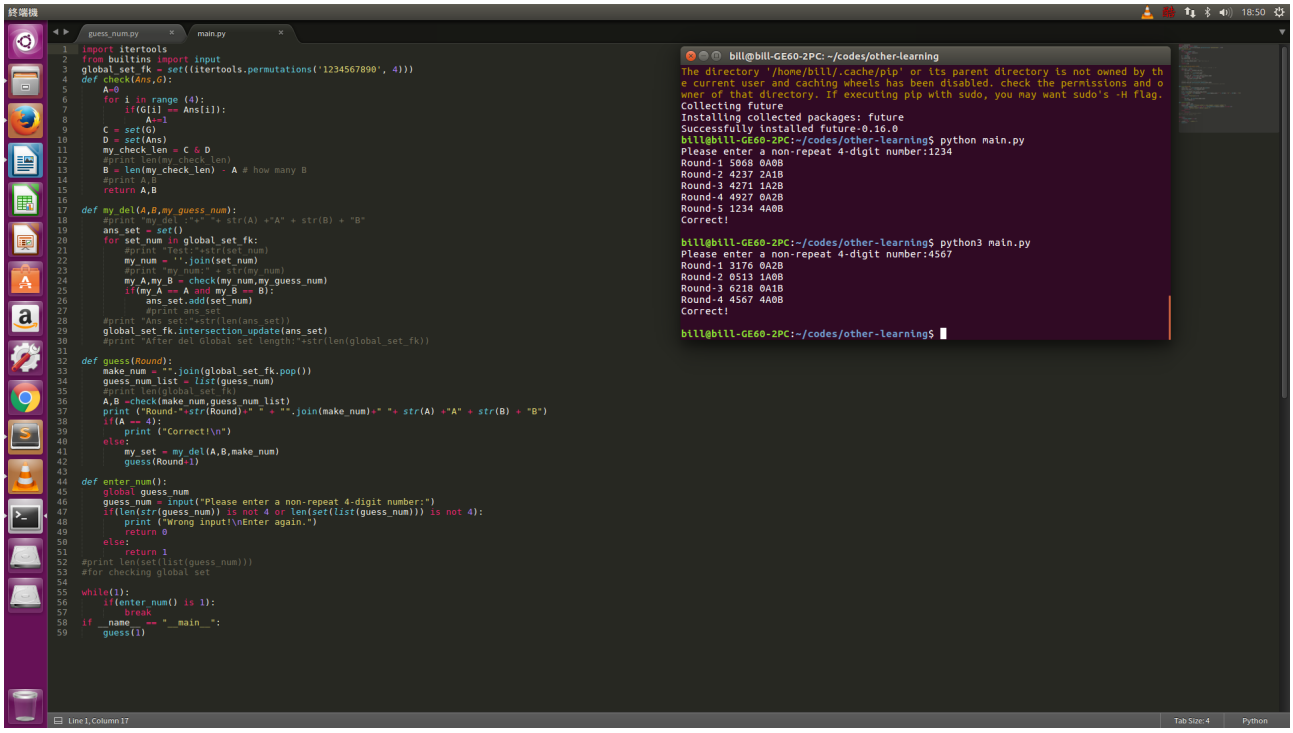
密碼學第三次作業

103062210 吳東翰



輸入 python 或是 python3 guess.py 執行(如果沒有，須先安裝，且需要 future、itertools 套件)
輸入 4 個不重複的數字，並開始執行



程式說明：

```
global_set_fk = set((itertools.permutations('1234567890', 4)))
```

global_set_fk 是所有 0-9 的排列組合，一開始共有 5040 種(10*9*8*7)。

```
def check(Ans,G):
	A=0
	for i in range (4):
		if(G[i] == Ans[i]):
			A+=1
	C = set(G)
	D = set(Ans)
	my_check_len = C & D
	#print len(my_check_len)
	B = len(my_check_len) - A # how many B
	#print A,B
	return A,B
```

check 是確認 Ans、G 有幾 A 幾 B 的關係。

```
def my_del(A,B,my_guess_num):
	#print "my_del :"+" "+ str(A) +"A" + str(B) + "B"
	ans_set = set()
	for set_num in global_set_fk:
		#print "Test:"+str(set_num)
		my_num = ''.join(set_num)
		#print "my_num:" + str(my_num)
		my_A,my_B = check(my_num,my_guess_num)
		if(my_A == A and my_B == B):
			ans_set.add(set_num)
			#print ans_set
	#print "Ans set:"+str(len(ans_set))
	global_set_fk.intersection_update(ans_set)
	#print "After del Global set length:"+str(len(global_set_fk))
```

my_del 是用來刪去 global_set_fk 中和 my_guess_num 不同幾 A 幾 B 的關係的函式。

```
def guess(Round):
	make_num = ''.join(global_set_fk.pop())
	guess_num_list = list(guess_num)
	#print len(global_set_fk)
	A,B =check(make_num,guess_num_list)
	print "Round-"+str(Round)+" " + ''.join(make_num)+" "+ str(A) +"A" + str(B) + "B"
	if(A == 4):
		print "Correct!\n"
	else:
		my_set = my_del(A,B,make_num)
		guess(Round+1)
```

guess()，首先先從 global_set_fk 中 pop 一個數字，並用他當作猜測的數字，並確認和 guess_num 之間幾 A 幾 B 的關係的關係，如果回傳的 A 是 4，即得到答案，如果不是，則刪去 global_set_fk 中和 make_num 不同幾 A 幾 B 的。

想法：

當程式首次猜對方的數字，並且得到回應之後，就可以依照所得的回應從 5040 個答案中刪除不可能的答案。程式先猜測得到一個回應後，即可把 5040 個答案中和猜測不同幾 A 幾 B 關係的答案刪掉，因正確答案的幾 A 幾 B 關係已經得到了，一定要只留下答案中幾 A 幾 B 關係和猜測所得到的一樣。如果猜測所得的是 1A2B，在原本 5040 個可能的答案中，只有 4*3*3*6=216 個符合這樣的描述。依照類似的方法，我們可以推算出，如果你猜的第一個數字得到 1A3B 和 0A0B 時，可能的答案的個數就分別變成 8 個和 360 個了。在執行數次之後，便能得到刪除不可能的候選答案，保留可能的答案，最後找出正確答案。

用 check_guess.py 來保證，所有的猜測，都會在 7 次以內結束。