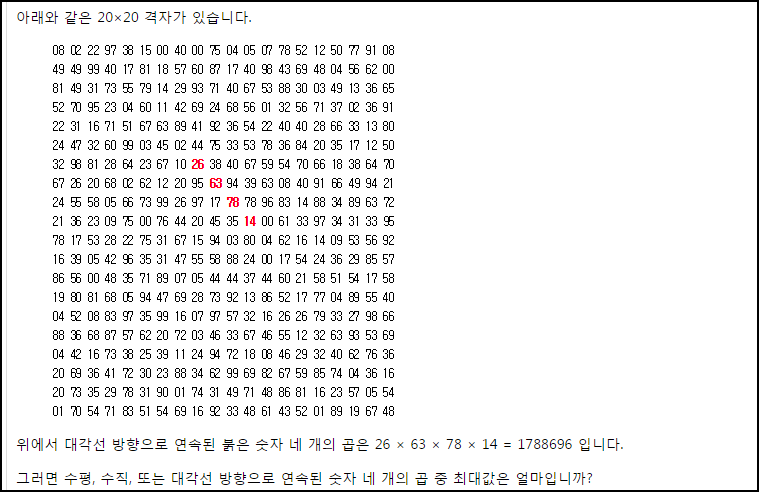
**Exercise**

20 x 20 격자의 수 집합에서 수평, 수직, 대각으로 연속하는 4개의 수의 곱들 중 최댓 값을 구하는 문제입니다.



**Solution\_code**

20x20 격자를 python list 로 만든 뒤에, 각각의 경우의 함수를 구현했습니다. 일관성 있는 규칙들이 각 경우 마다 있으므로, check 할 수 있었습니다.

|  |  |  |
| --- | --- | --- |
| **1**  **2**  **3**  **4**  **5**  **6**  **7**  **8**  **9**  **10**  **11**  **12**  **13**  **14**  **15**  **16**  **17**  **18**  **19**  **20**  **21**  **22**  **23**  **24**  **25**  **26**  **27**  **28**  **29**  **30**  **31**  **32**  **33**  **34**  **35**  **36**  **37**  **38**  **39**  **40**  **41**  **42**  **43**  **44**  **45**  **46**  **47**  **48**  **49**  **50**  **51**  **52**  **53**  **54**  **55**  **56**  **57**  **58**  **59**  **60**  **61**  **62**  **63**  **6** | **numb = [8,2,22,97,38,15,0,40,0,75,4,5,7,78,52,12,50,77,91,8,49,49,99,40,17,81,18,57,60,87,17,40,98,43,69,48,4,56,62,0,81,49,31,73,55,79,14,29,93,71,40,67,53,88,30,3,49,13,36,65,52,70,95,23,4,60,11,42,69,24,68,56,1,32,56,71,37,2,36,91,22,31,16,71,51,67,63,89,41,92,36,54,22,40,40,28,66,33,13,80,24,47,32,60,99,3,45,2,44,75,33,53,78,36,84,20,35,17,12,50,32,98,81,28,64,23,67,10,26,38,40,67,59,54,70,66,18,38,64,70,67,26,20,68,2,62,12,20,95,63,94,39,63,8,40,91,66,49,94,21,24,55,58,5,66,73,99,26,97,17,78,78,96,83,14,88,34,89,63,72,21,36,23,9,75,0,76,44,20,45,35,14,0,61,33,97,34,31,33,95,78,17,53,28,22,75,31,67,15,94,3,80,4,62,16,14,9,53,56,92,16,39,5,42,96,35,31,47,55,58,88,24,0,17,54,24,36,29,85,57,86,56,0,48,35,71,89,7,5,44,44,37,44,60,21,58,51,54,17,58,19,80,81,68,5,94,47,69,28,73,92,13,86,52,17,77,4,89,55,40,4,52,8,83,97,35,99,16,7,97,57,32,16,26,26,79,33,27,98,66,88,36,68,87,57,62,20,72,3,46,33,67,46,55,12,32,63,93,53,69,4,42,16,73,38,25,39,11,24,94,72,18,8,46,29,32,40,62,76,36,20,69,36,41,72,30,23,88,34,62,99,69,82,67,59,85,74,4,36,16,20,73,35,29,78,31,90,1,74,31,49,71,48,86,81,16,23,57,5,54,1,70,54,71,83,51,54,69,16,92,33,48,61,43,52,1,89,19,67,48]**    **def Diagonal(numb):**  **temp = []**  **num\_mod = 1**  **cnt = 1**  **j =0**  **while(cnt != 18):**  **for i in range(j,j+17):**  **# print numb[i], numb[i+21], numb[i+21\*2], numb[i+21\*3]**  **num\_mod \*= numb[i]**  **num\_mod \*= numb[i+21]**  **num\_mod \*= numb[i+21\*2]**  **num\_mod \*= numb[i+21\*3]**    **temp.append(num\_mod)**  **num\_mod = 1**  **j += 20**  **cnt += 1**    **temp = list(set(temp))**  **temp.sort()**  **return max(temp)**      **def Row\_mod(numb):**  **temp = []**  **num\_mod = 1**  **cnt = 1**  **j =0**  **while(cnt != 21):**  **for i in range(j,j+17):**  **# print numb[i], numb[i+1], numb[i+2], numb[i+3]**  **num\_mod \*= numb[i]**  **num\_mod \*= numb[i+1]**  **num\_mod \*= numb[i+2]**  **num\_mod \*= numb[i+3]**    **temp.append(num\_mod)**    **num\_mod = 1**    **j += 20**  **cnt += 1**    **temp = list(set(temp))**  **temp.sort()**  **return max(temp)**      **def Col\_mod(numb):**  **temp = []**  **num\_mod = 1**  **cnt = 1**  **j =0**  **while(cnt != 18):**  **for i in range(j,j+20):**  **# print numb[i], numb[i+20], numb[i+20\*2], numb[i+20\*3]**  **num\_mod \*= numb[i]**  **num\_mod \*= numb[i+20]**  **num\_mod \*= numb[i+20\*2]**  **num\_mod \*= numb[i+20\*3]**    **temp.append(num\_mod)**    **num\_mod = 1**    **j += 20**  **cnt += 1**    **temp = list(set(temp))**  **temp.sort()**    **return max(temp)**      **def Reverse\_Diagonal(numb):**  **temp = []**  **num\_mod = 1**  **cnt = 1**  **j =0**  **while(cnt != 18):**  **for i in range(j+3,j+20):**  **# print numb[i], numb[i+20-1], numb[i+20\*2-2], numb[i+20\*3-3]**  **num\_mod \*= numb[i]**  **num\_mod \*= numb[i+20-1]**  **num\_mod \*= numb[i+20\*2-2]**  **num\_mod \*= numb[i+20\*3-3]**    **temp.append(num\_mod)**    **num\_mod = 1**    **j += 20**  **cnt += 1**    **temp = list(set(temp))**  **temp.sort()**  **return max(temp)**      **print "[+] Diagonal : ", Diagonal(numb)**  **print "[+] Row : ",Row\_mod(numb)**  **print "[+] Col : ",Col\_mod(numb)**  **print "[+] R\_Diagonal : ",Reverse\_Diagonal(numb)**  ***[Colored by Color Scripter](http://colorscripter.com/info#e)*** | [cs](http://colorscripter.com/info#e) |

**Result**

