به نام خدا

برنامه نویسی پیشرفته(**AP**)
گزارش تمرین سری پنجم

امیرحسین زمانی-۹۴۲۳۸۰۲
۱۳۹۸/۰۳/۱۰

لینک این سری از تمرین بر روی **github** :

https://github.com/AHHHZ975/AP-HW5

سوال ۱ :

آ) با استفاده از تابع std::iota در کتابخانه numeric ، اینکار به سادگی و به صورت زیر انجام شد:

```
    vector<int> vec1(100);
    vector<int> vec2(10);
    iota(begin(vec1), end(vec1), 1);
    iota(begin(vec2), end(vec2), 1);
```

ب) با استفاده از دستور insert به راحتی وکتور دوم به انتهای وکتور اول اضافه شد:

```
    vec1.insert(vec1.end(), vec2.begin(), vec2.end());
    std::copy(vec1.begin(), vec1.end(), out_it);
    cout << endl;
```

```
ahz@AHZ:~/Desktop/AP-HW5-9423802/Q1$ ./main
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 5
7 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83
84 85 86 87 88 89 90 91 92 93 94 95 96 97 98 99 100 1 2 3 4 5 6 7 8 9 10
```

ج) با استفاده از دستور copy_if همانند تمرین سری قبل داریم:

```
vector<int> odd_vec(vec1.size()/2);
copy_if(vec1.begin(), vec1.end(), odd_vec.begin(), [](int i){return i%2==1;});
std::copy(odd_vec.begin(), odd_vec.end(), out_it);
cout << endl;
```

```
.ahz@AHZ:~/Desktop/AP-HW5-9423802/Q1$ ./main
1 3 5 7 9 11 13 15 17 19 21 23 25 27 29 31 33 35 37 39 41 43 45 47 49 51 53 55 5
7 59 61 63 65 67 69 71 73 75 77 79 81 83 85 87 89 91 93 95 97 99
```

د) برای این قسمت دو راه وجود دارد. راه اول استفاده از دستور reverse_copy است:

```
    vector<int> reverse_vec;
    std::reverse_copy(vec1.begin(), vec1.end(), std::back_inserter(reverse_vec));
    std::copy(reverse_vec.begin(), reverse_vec.end(), out_it);
    cout << endl;
```

```
.ahz@AHZ:~/Desktop/AP-HW5-9423802/Q1$ ./main
100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 7
4 73 72 71 70 69 68 67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48
47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21
 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
```

و راه دوم استفاده از دستور insert است به طوری که از iterator های rbegin و rend استفاده شود. این دو، reverse iterator های یک وکتور را در اختیار می گذارند.

```
    vector<int> reverse_vec;
    reverse_vec.insert(reverse_vec.begin(), vec1.rbegin(), vec1.rend());
    std::copy(reverse_vec.begin(), reverse_vec.end(), out_it);
    cout << endl;
```

```
.ahz@AHZ:~/Desktop/AP-HW5-9423802/Q1$ ./main
100 99 98 97 96 95 94 93 92 91 90 89 88 87 86 85 84 83 82 81 80 79 78 77 76 75 7
4 73 72 71 70 69 68 67 66 65 64 63 62 61 60 59 58 57 56 55 54 53 52 51 50 49 48
47 46 45 44 43 42 41 40 39 38 37 36 35 34 33 32 31 30 29 28 27 26 25 24 23 22 21
 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
```

ه) به صورت عادی:

```
sort(vec2.begin(), vec2.end());
std::copy(vec2.begin(), vec2.end(), out_it);
cout << endl;
```

برای حالت موازی و بر روی linux و کامپایلر GNU و GCC ، استفاده از std::execution::par منجر به ارور در کامپایلر می شد که با جستجو در اینترنت ظاهرا هنوز روی این کامپایلرها این تابع در STL اضافه نشده است و درنتیجه کامپایلر ارور زیر را نشان می دهد.

```
ahz@AHZ:~/Desktop/AP-HW5-9423802/Q1$ make
g++ -std=c++17 -Wall -c main.cpp
main.cpp: In function ‘int main()’:
main.cpp:35:12: error: ‘std::execution’ has not been declared
  sort(std::execution::par, vec2.begin(), vec2.end());
            ^~~~~~~~~
Makefile:11: recipe for target 'main.o' failed
make: *** [main.o] Error 1
```

حالت parallel بر روی visual studio در ویندوز هم تست شد اما متاسفانه با Errorی مشابه فوق مواجه شدم.

سوال ۲ :

آرگومان args* این امکان را می دهد که می توان هرتعداد متغیری را که کاربر نیاز دارد بدون آنکه از قبل آن ها را تعریف کرده باشد، به تابع پاس دهد و از آن ها استفاده کند و البته این متغیرها می توانند non-keyworded باشند یعنی خود متغیر نام نداشته باشد و عملا به صورت موقتی یا Temporary به تابع پاس داده شود. به عنوان مثال:

```python
def testArgv(*argv):
    for arg in argv:
        print(arg)


testArgv('Hello', 'Welcome', 'to', 'AP', 'calss')
```

```
ahz@AHZ:~/Desktop/AP-HW5-9423802/Q2$ python3 main.py
Hello
Welcome
to
AP
calss
```

آرگومان keyword** هم برای پاس دادن هر تعداد از آرگومان هایی که دارای نام هستند یعنی به اصطلاح Keyworded هستند استفاده می شود.

```python
def testKeyword(**kwargs):
    for key, value in kwargs.items():
        print ("%s == %s" %(key, value))


testKeyword(first ='Hello', mid ='AP', last='Students')
```

```
first == Hello
mid == AP
last == Students
```

سوال ۳ :

```
A0 = dict(zip(('a', 'b', 'c', 'd', 'e'), ('1', '2', '3', '4', '5')))
A1 = range(10)
A2 = [i for i in A1 if i in A0]
A3 = sorted(A0[i] for i in A0)
A4 = [[i, i*i] for i in A1]
```

الف)

A۰ یک دیکشنری با مقادیر روبه روست: {'a': '۱', 'b': '۲', 'c': '۳', 'd': '۴', 'e': '۵'}

A۱ یک لیست از اعداد ۰ تا ۹ است.(خود اعداد ۰ و ۹ در این لیست حضور دارند)

A۲ یک لیست خالی است.

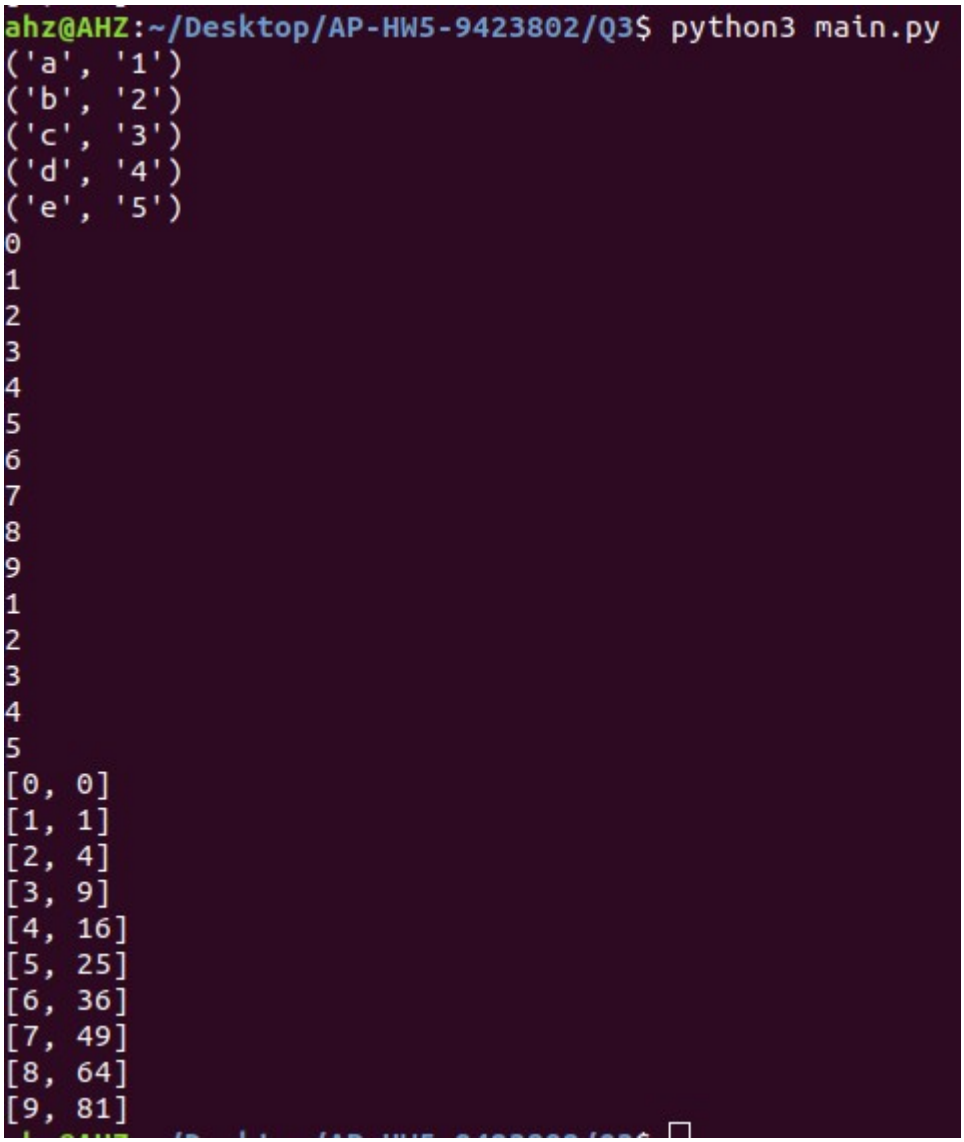
A۳ یک لیست مرتب شده از کوچک به بزرگ، از مقادیر دیکشنری A۰ است. یعنی: [۱, ۲, ۳, ۴, ۵]

A۴ یک لیست از مجموعه لیست های دوتایی است که عنصر دوم هرکدام از این لیست هاُ، مجذور عنصر اول است. عنصر اول هم از ۰ تا ۹ است. یعنی: [[۰, ۰], [۱, ۱], [۲, ۴], [۳, ۹], [۴, ۱۶], [۵, ۲۵], [۶, ۳۶], [۷, ۴۹], [۸, ۶۴], [۹, ۸۱]]

ب) با extend کردن یک لیست دیگر به نام A۵ به راحتی می توان همه مقادیر این ۴ متغیر را در یک حلقه نشان داد.

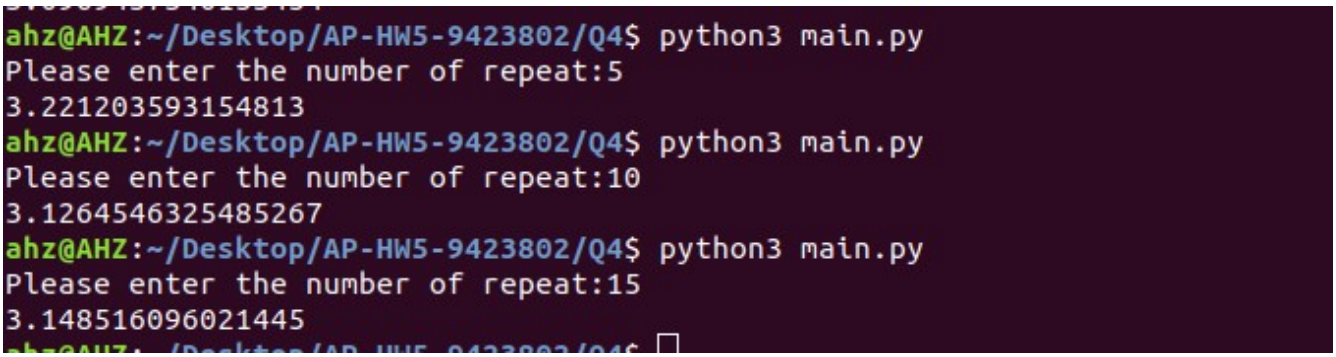
```
A0 = dict(zip(('a', 'b', 'c', 'd', 'e'), ('1', '2', '3', '4', '5')))
A1 = range(10)
A2 = [i for i in A1 if i in A0]
A3 = sorted(A0[i] for i in A0)
A4 = [[i, i*i] for i in A1]

A0 = A0.items()
A5 = []
A5.extend(A0)
A5.extend(A1)
A5.extend(A2)
A5.extend(A3)
A5.extend(A4)
for item in A5:
    print(item)
```



سوال ۴:

دقیقا همانند توضیحات سوال، کد پیاده سازی شد و درنهایت نتایج زیر به طور مثال برای تعداد تکرار ۵ ، ۱۰ و ۱۵ حاصل شدند:



سوال ۵:

```python
import os

def create_dir(name, address):
    directory = str(address) + '/' + str(name)
    if not os.path.exists(directory):
        os.makedirs(directory)


def create_file(name, address):
    directory = str(address)
    if not os.path.exists(directory):
        os.makedirs(directory)
    with open(os.path.join(directory, str(name)), 'w'):
        pass

def delete(name, address):
    directory = str(address) + '/' + str(name)
    if os.path.isfile(directory):
        os.remove(directory)

def find(name, address):
    for root, dirs, files in os.walk(str(address)):
        for file in files:
            if file.endswith(str(name)):
                print(os.path.join(root, file))

create_dir('ahz', 'ahz')
create_file('ahz.txt', 'ahz')
delete('ahz.txt', 'ahz')
find('ahz.txt', 'ahz')
```

سوال ۶:

```
ahz@AHZ:~/Desktop/AP-HW5-9423802/Q6$ python3 main.py
Result of python code (n = 2): 0.15749282438578963
Execution time of Python code: 0.00029277801513671875 seconds
Result of C++ code (n =  2): 0.1153386257122344501
Execution time of C++ code: 0.006436347961425781 seconds



Result of C++ code (n =  1): 0.080742701809220398685
Result of C++ code (n =  2): 0.1153386257122344501
Result of C++ code (n =  3): 0.10880746833884317331
Result of C++ code (n =  4): 0.10875598770543667561
Result of C++ code (n =  5): 0.1087619580095648715
Result of C++ code (n =  6): 0.10876191480545721316
Result of C++ code (n =  7): 0.10876191279387334492
Result of C++ code (n =  8): 0.10876191279529763416
Result of C++ code (n =  9): 0.10876191279516668794
Result of C++ code (n = 10): 0.1087619127951560563



  N  Python time    C++ time
---  ------------  ----------
  1  0.000058 s    0.006582 s
  2  0.000264 s    0.002147 s
  3  0.000295 s    0.002325 s
  4  0.001446 s    0.002244 s
  5  0.001267 s    0.003781 s
  6  0.006326 s    0.002821 s
  7  0.005671 s    0.002950 s
  8  0.010350 s    0.002209 s
  9  0.008141 s    0.003215 s
 10  0.033471 s    0.002907 s
```

سوال ۷:

با استفاده از list comprehension به سادگی نتیجه زیر برای سه ورودی حاصل شد:

input = [۱, ۲, ۳, ۴, ۵, ۶, ۷, ۸, ۹, ۱۰, ۱۱, ۱۲, ۱۳, ۱۴, ۱۵, ۱۶, ۱۷, ۱۲]

```
ahz@AHZ:~/Desktop/AP-HW5-9423802/Q7$ python3 main.py
The result is: 6 12
```

input = [۳ ۴ ۱ ۳۷ ۲۱ ۱۸ ۲۳ ۲۱ ۲۷ ۲۲ ۴۳ ۲۱]

```
ahz@AHZ:~/Desktop/AP-HW5-9423802/Q7$ python3 main.py
The result is: 18
```

input = [۱, ۲, ۳, ۴, ۵, ۶, ۷, ۸, ۹, ۱۰, ۱۱, ۱۲]

```
ahz@AHZ:~/Desktop/AP-HW5-9423802/Q7$ python3 main.py
The result is: 6 12
```