What is the output of the following code snippet?

```
for number in 10,15:
    for counter in range(1,3):
        print(number*counter, end=" ")
```

- 10 20 30 15 30 45
- 10 15 20 30 30 45
- 10 15 20 30
- 10 20 15 30

Which of the following statements may result in an error?

(Assume that the statements are executed in the order in which they are written.)

a. tuple1=(5,10,15,20,25)
b. print(len(tuple1))
c. tuple1[2]=100
d. print(tuple1[5])
e. tuple1=tuple1+(8,9,"h")

- d
- e
- a and d
- c and d

What is the output of the following code snippet?

```
my_list=[0]*5
for index in range(1,5):
    my_list[index]=(index-1)*10
print(my_list)
```

- [0, 0, 0, 0, 0]
- [10, 20, 30, 40, 50]
- [0, 0, 10, 20, 30]
- [0, 10, 20, 30, 40]



A developer wants to use a Python data type in which they can store the element value as well as its index. Which of the following data types can they use to do so?

- Tuple
- List
- big Int
- Dict

What is the output of the given code?

```
import re
m = re.search('a%^& b%', 'abcdefghijklmnopqrstuvwxyz')
m.group(0)
print(m)
```

- ab
- abcdef
- djdie
- Error

What is the output of the given code snippet?

```
Sample = ((1, 'KKK', 200), (307, 'LLL', 81), (56, 'MMM', 9))
res = [{'key': sub[0], 'value': sub[1], 'id': sub[2]} for sub in Sample]
print(str(res))
```

- [{'key': 200, 'value': 'KKK', 'id': 1}, {'key': 81, 'value': 'LLL', 'id': 307}, {'key': 9, 'value': 'MMM', 'id': 56}]
- [{'key': 1, 'value': 'KKK', 'id': 200}, {'key': 307, 'value': 'LLL', 'id': 81}, {'key': 56, 'value': 'MMM', 'id': 9}]
- [{'key': 1, 'value': 'KKK', 'id': 200}, {'key': 307, 'value': 'LLL', 'id': 81}, {'key': 56, 'value': 'MM', 'id': 9}]
- Error

What is the output of the given code?

```
a={1,"hello",2,"World",3,True}
for i in range(4):
    a.add(i)
print(a)
```

- {1,"hello",2,"World",3,True,0,1,2,3}
- {1,"hello",2,"World",3,True,0}
- {0, 1, 2 ,3, 1, 2, 3, 'World', 'hello'}
- {0, 1, 2, 3, 'World', 'hello'}

What is the output of the given code?

```
mydict = {}
mydict[1]=2
mydict[2]=3
mydict[3]=4

i = 5
while i>=0:
  mydict[i]=i%3
  i-=1
print(len(mydict), mydict[2])
```

- 44684
- 44683
- 44654
- 44714

A machine learning engineer has a data set with four columns, namely flat no., area, price, and property. They want to change the column name flat no. with house no. and price with a value. Which of the following codes is correct for this purpose?

- `df = pd.read_csv('data.csv')`
  `df = df.rename(columns={"flatno":"houseno","price":"value"})`
- `df = pd.read_csv('data.csv')`
  `df = df.rename(column={"flatno":"houseno","price":"value"})`
- `df = pd.read_csv('data.csv')`
  `df = df.rename(column={"flatno":"houseno"},column={"price":"value"})`
- `"df = pd.read_csv('data.csv')`
  `df = df.chnagename(column={""flatno"":""houseno""},column={""price"":""value""})"`



What is the output of the given code?

```
FHW=open("data.txt","w")
FHW.write("written something")
print(FHW.tell())
print("closed?",FHW.closed)
FHW.close()
print("after closing the file closed?",FHW.closed)
```

- 17
  closed? False
  after closing the file closed? True
- 17
  closed? True
  after closing the file closed? False
- 18
  closed? False
  after closing the file closed? True
- 18
  closed? True
  after closing the file closed? False

What should be the value of num1 and num2 to get the output "1"?

```
if((num1/num2==5) and (num1+num2)>5):
    print("1")
elif((num1-num2)<=1 or (num1%num2)==0):
    print("2")
else:
    print("3")
```

- num1=11, num2=2
- num1=0, num2=5
- num1=5, num2=1
- num1=-10,num2=2

Which of the following statements may result in an error?

(Assume that the statements are executed in the order in which they are written.)

a. list1=[5,10,15,20,25]
b. print(len(list1))
c. print(list1[4])
d. print(list1[5])
e. print(list1[4:5])
f. list1[2]=12
g. print(list1)
h. list1=list1+[8,9]

- b
- d
- e
- h

Which line in the following code snippet has an error?

```
def myfun():
 print("")
 def myinnerfun(): # line 1
 var = 10 # line 2
 print(var) # line 3
 myinnerfun() # line 4
 print(var) # line 5
myfun()
```

- Line 1
- Line 5
- Line 4
- Line 2



What is the output of the following code if the input is 5 4?

```
def power(x, n):
l=[i for i in range(1,x+1)]
return sum(l)+n
if __name__ == '__main__':
first_multiple_input = input().rstrip().split()
 x = int(first_multiple_input[0])
 n = int(first_multiple_input[1])
 result = power(x, n)
 print(result)
```

- 19
- 33
- 15256
- Syntax error

You have a function named "clear" whose implementation is shown as follows.

```
def clear(string):

 pattern = '\s+'
 new_string = re.subn(pattern, '', string)
 return new_string
```

What kind of string is the clear function returning here?

A) ENTER OPTION
B) It will clear all the new lines from the string
C) It will remove all the white space characters from the string

- A
- B
- C
- A and C

A developer working on Collection(list) writes the following code. What is the output produced by the code?

```
my_list = [0] * 5
for index in range(1, 5):
    my_list[index] = (index - 1) * index

print(my_list)

list1 = [10, 20, 30, 40, 50]
list1.insert(7, 25)
print(list1)
```

- [0, 0, 2, 6, 12] [10, 20, 30, 40, 50, 25]
- [0, 2, 6, 12] [20, 30, 40, 50, 25]
- [0, 7, 2, 6, 12] [10, 20, 30, 40, 50, 25]
- Error

A developer working on dictionary comprehension writes the following code. What is the output produced by this code?

```
list = [1, 2, 3, 4, 5, 6, 7,9,11,13]

dict = {}

for var in list:
    if var % 2 != 0:
        dict[var] = var ** 3
    else:
        dict[var] = var ** 4

print(dict)
```

- {1: 1, 2: 16, 3: 27, 4: 256, 5: 125, 6: 1296, 7: 343, 9: 729, 11: 1331, 13: 2197}
- {1: 1, 2: 8, 3: 81, 4: 64, 5: 625, 6: 216, 7: 2401, 9: 6561, 11: 14641, 13: 28561}
- Compilation error
- Runtime error



What is the output of the given code?

```
def find_sum(a,b):
    try:
        print(a+c)
    except NameError:
        print("Function name error")
    finally:
        print("Sum finally")
try:
    find_sum(12,13)
except NameError:
    print("Invocation name error")
finally:
    print("Invocation finally")
```

- Function name error
  Sum finally
- Function name error
  Sum finally
  Invocation finally
- Function name error
  Sum finally
  Invocation name error
  Invocation finally
- Sum finally
  Invocation name error
  Invocation finally

Conor is working on a NumPy array and is performing some basic mathematical operations on it to write the following code.

```
import NumPy as np

a = np.array([7,3,4,5,1])
b = np.array([3,4,5,6,7])

print(a[a%b[1]][2]+b[a%b[2]][4])
```

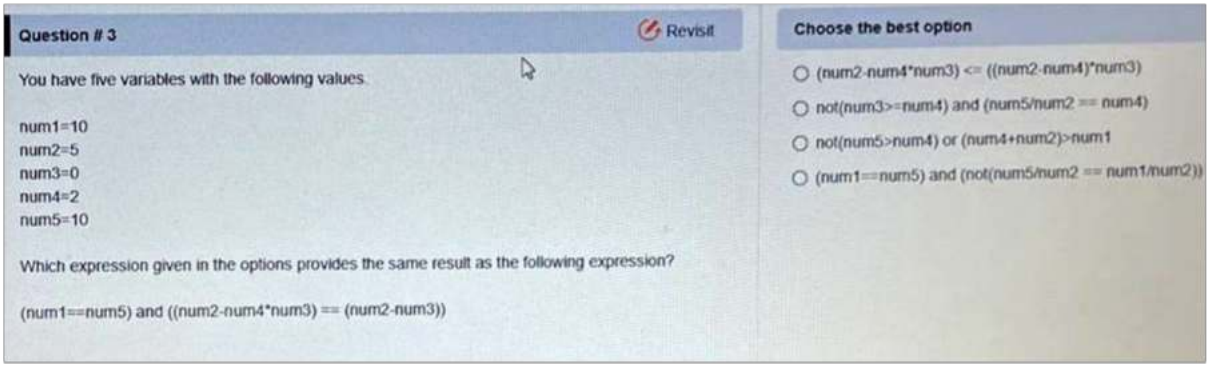
What is the output produced by the given code?

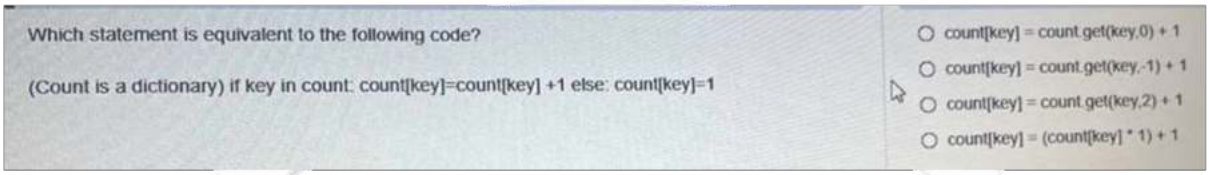
- 11
- [11]
- [6]
- 6

What is the output of the given code snippet?

```
A = ['XZ', 'TV','ABCD','123']
print(list(map(list, A)))
```

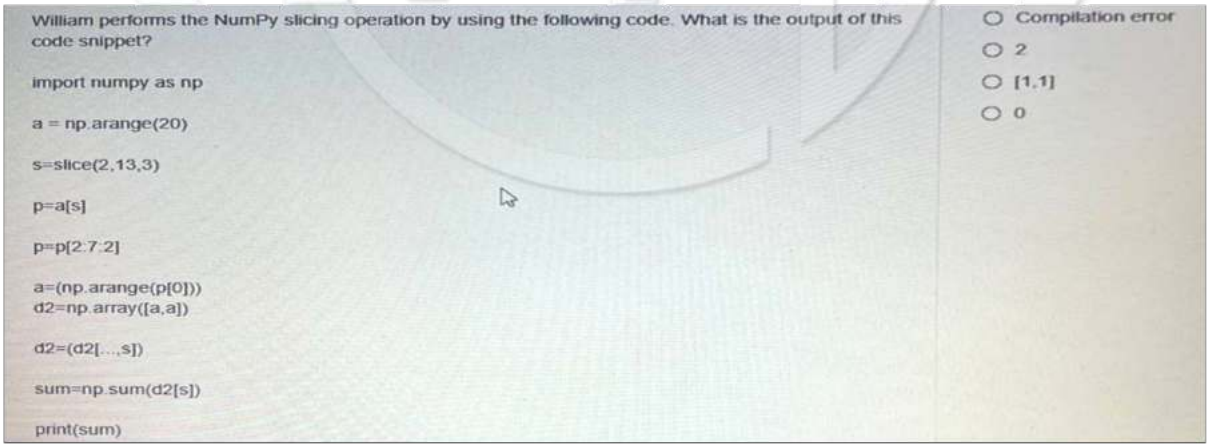
- [['X', 'Z'], ['T, 'V'], ['A', 'B', 'C', 'D'], ['1', '2', '3']]
- [['XZ'], ['TV], ['ABCD'], ['123']]
- [['X,Z'], ['T','V'], ['ABCD'], ['123']]
- [['XZ'], ['TV], ['A,B,C,D'], ['1,2,3']]









What is the output of the following code snippet?

```
message="welcome to New_York"
word=message[-7:]
if(word=="New York"):
   print("got it")
else:
   message=message[3:14]
   print(message)
```

- come to New
- got it
- kroY weN
- lcome to New_

Which of the following options explains what stride (2, 20) means?

- Proceed 20 bits to process to the next column and 2 bits to locate the next row.
- Proceed 2 bits to the next column and 20 bits to locate the next row.
- Proceed 20 bytes to the next column and 2 bytes to locate the next row.
- Proceed 2 bytes to the next column and 20 bytes to locate the next row.

What is the output of the given code?

```
def find_sum(a,b):
   try:
      print(a+c)
   except ValueError:
      print("Function name error")
   finally:
      print("Sum finally")
try:
   find_sum(12,13)
except NameError:
   print("Invocation name error")
finally:
   print("Invocation finally")
```

- Sum finally
  Invocation name error
  Invocation finally
- Function name error
  Sum finally
  Invocation name error
  Invocation finally
- Function name error
  Sum finally
  Invocation finally
- Function name error
  Sum finally



As a developer, Ronaldo wrote the following code with a missing line. What should be written in line 12 to get the output 6?

```
class Example:
   def __init__(self):
      self.__num=5

   def set_num(self,num):
      self.__num=num

   def get_num(self):
      return self.__num
obj=Example()
obj.set_num(6)
#line 12
```

- print(obj.__num)
- print(obj.get_num())
- print(obj.num)
- print(num)

What is the output of the given code snippet based on the OOPs concept?

```
class Example:
 def __init__(self,num):
 self.num=num

 def set_num(self,num):
 self.num=num

 def get_num(self):
 return self.num
obj=Example(10)
print(obj.get_num())
obj.set_num(15)
print(obj.get_num())
```

- 44849
- 15 10
- 44844
- 15 15

What is the output of the given code?

```
def fn(x):
 for i in range(1,x):
 if(i%3==0):
 break
 elif(x%2==0):
 continue
 else:
 del i
 print(i)
 x-=i

fn(10)
```

- 0
- 2
- 3
- Error

A developer working on dictionary comprehension writes the following code. What is the output produced by this code?

```
list = [1, 2, 3, 4, 5, 6, 7,9,11,13]

dict = {}

for var in list:
    if var % 2 != 0:
        dict[var] = var ** 3
    else:
        dict[var] = var ** 4

print(dict)
```

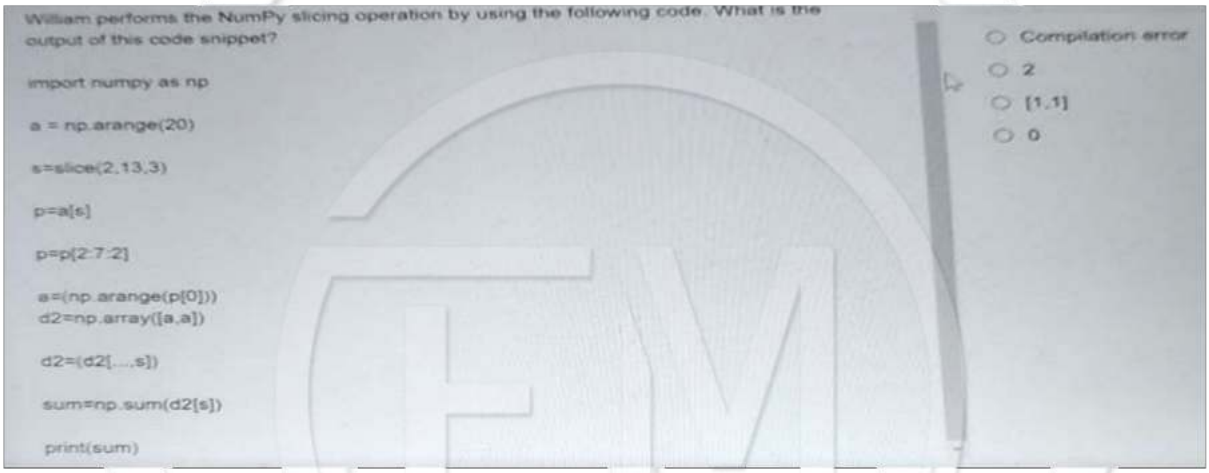
- {1: 1, 2: 16, 3: 27, 4: 256, 5: 125, 6: 1296, 7: 343, 9: 729, 11: 1331, 13: 2197}
- {1: 1, 2: 8, 3: 81, 4: 64, 5: 625, 6: 216, 7: 2401, 9: 6561, 11: 14641, 13: 28561}
- Compilation error
- Runtime error

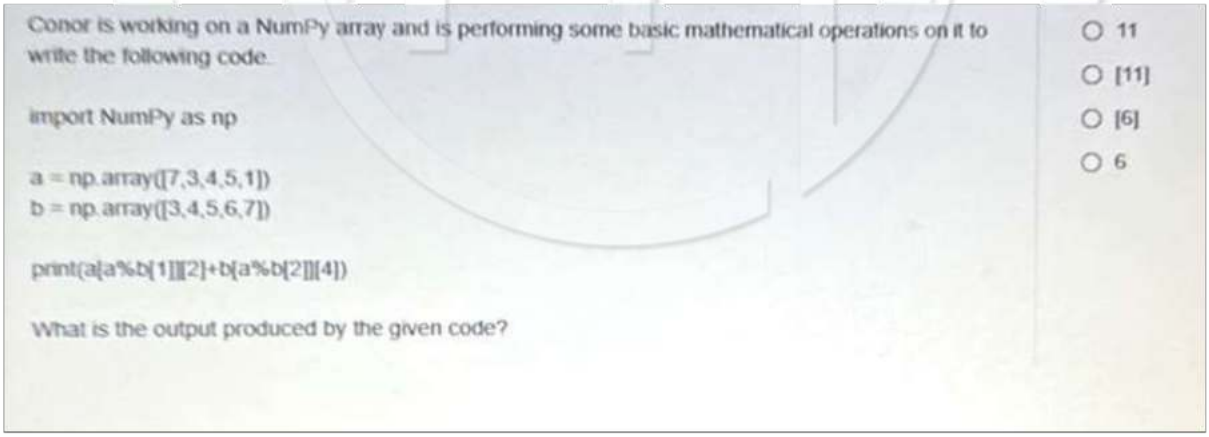
What is the output of the given code snippet based on the OOPs concept?

```
class Example:
 def __init__(self,num):
 self.num=num

 def set_num(self,num):
 self.num=num

 def get_num(self):
 return self.num
obj=Example(10)
print(obj.get_num())
obj.set_num(15)
print(obj.get_num())
```

- 44849
- 15 10
- 44844
- 15 15



A developer working on Collection(list) writes the following code. What is the output produced by the code?

```
my_list = [0] * 5
for index in range(1, 5):
    my_list[index] = (index - 1) * index

print(my_list)

list1 = [10, 20, 30, 40, 50]
list1.insert(7, 25)
print(list1)
```

- [0, 0, 2, 6, 12] [10, 20, 30, 40, 50, 25]
- [0, 2, 6, 12] [20, 30, 40, 50, 25}
- [0, 7, 2, 6, 12] [10, 20, 30, 40, 50, 25]
- Error

You have a function named "clear" whose implementation is shown as follows.

```
def clear(string):

    pattern = '\s+'
    new_string = re.subn(pattern, '', string)
    return new_string
```

What kind of string is the clear function returning here?

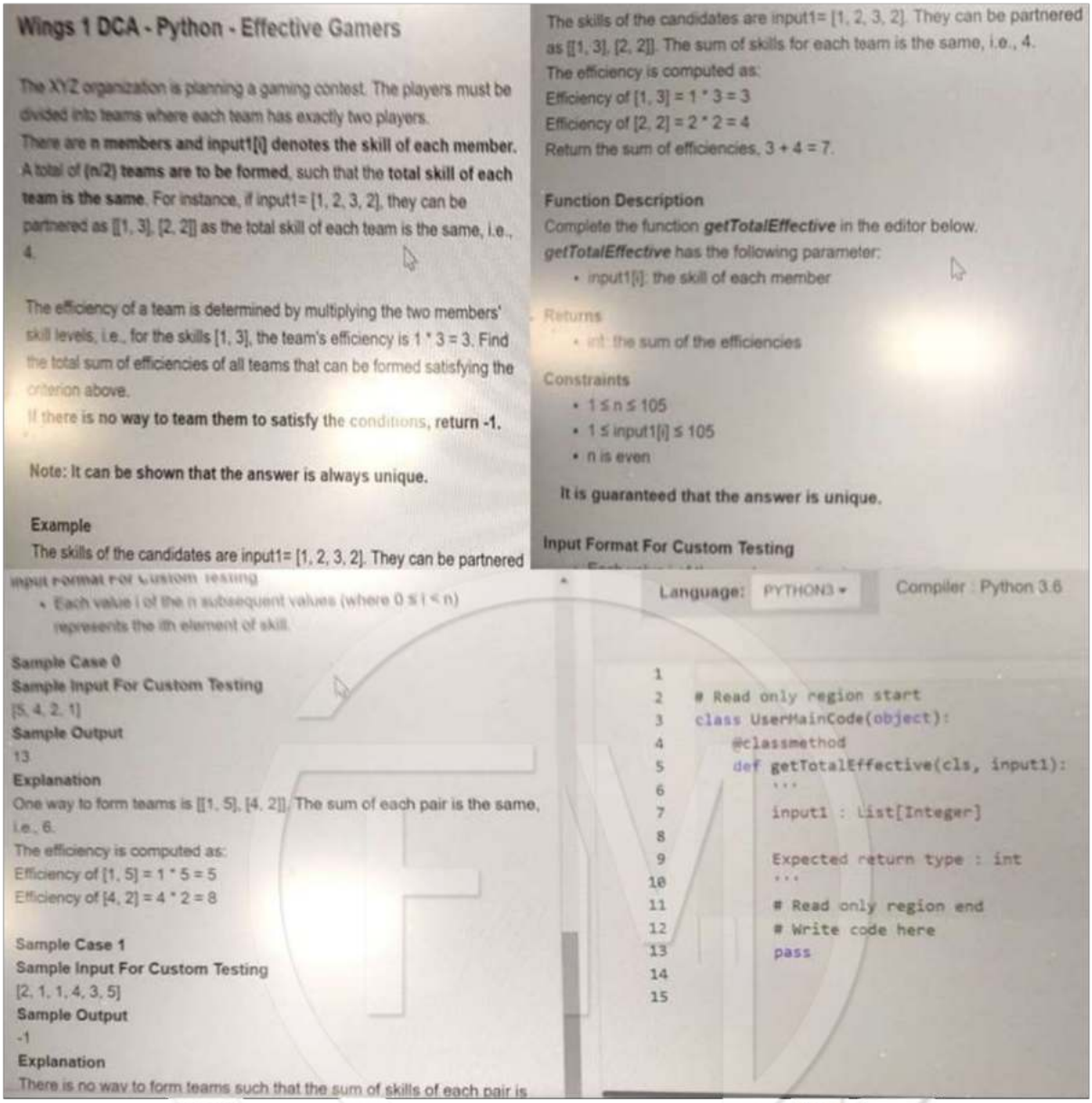
A) ENTER OPTION
B) It will clear all the new lines from the string
C) It will remove all the white space characters from the string

- A
- B
- C
- A and C

What is the output of the given code?

```
def find_sum(a,b):
    try:
        print(a+c)
    except NameError:
        print("Function name error")
    finally:
        print("Sum finally")
try:
    find_sum(12,13)
except NameError:
    print("Invocation name error")
finally:
    print("Invocation finally")
```

- Function name error Sum finally
- Function name error Sum finally Invocation finally
- Function name error Sum finally Invocation name error Invocation finally
- Sum finally Invocation name error Invocation finally



## Wings 1 DCA - Python - Effective Gamers

The XYZ organization is planning a gaming contest. The players must be divided into teams where each team has exactly two players.
There are **n members and input1[i] denotes the skill of each member.**
A total of **(n/2) teams are to be formed**, such that the **total skill of each team is the same**. For instance, if input1= [1, 2, 3, 2], they can be partnered as [[1, 3], [2, 2]] as the total skill of each team is the same, i.e., 4.

The efficiency of a team is determined by multiplying the two members' skill levels, i.e., for the skills [1, 3], the team's efficiency is 1 * 3 = 3. Find the total sum of efficiencies of all teams that can be formed satisfying the criterion above.
If there is **no way to team them to satisfy** the conditions, **return -1.**

Note: **It can be shown that the answer is always unique.**

**Example**
The skills of the candidates are input1= [1, 2, 3, 2]. They can be partnered as [[1, 3], [2, 2]]. The sum of skills for each team is the same, i.e., 4.
The efficiency is computed as:
Efficiency of [1, 3] = 1 * 3 = 3
Efficiency of [2, 2] = 2 * 2 = 4
Return the sum of efficiencies, 3 + 4 = 7.

**Function Description**
Complete the function ***getTotalEffective*** in the editor below.
***getTotalEffective*** has the following parameter:
- input1[i]: the skill of each member

Returns
- int: the sum of the efficiencies

Constraints
- $1 \le n \le 105$
- $1 \le input1[i] \le 105$
- n is even

**It is guaranteed that the answer is unique.**

**Input Format For Custom Testing**
- Each value i of the n subsequent values (where $0 \le i < n$) represents the ith element of skill.

**Sample Case 0**
**Sample Input For Custom Testing**
[5, 4, 2, 1]
**Sample Output**
13
**Explanation**
One way to form teams is [[1, 5], [4, 2]]. The sum of each pair is the same, i.e., 6.
The efficiency is computed as:
Efficiency of [1, 5] = 1 * 5 = 5
Efficiency of [4, 2] = 4 * 2 = 8

**Sample Case 1**
**Sample Input For Custom Testing**
[2, 1, 1, 4, 3, 5]
**Sample Output**
-1
**Explanation**
There is no way to form teams such that the sum of skills of each pair is

Language: PYTHON3 | Compiler : Python 3.6

```
# Read only region start
class UserMainCode(object):
    @classmethod
    def getTotalEffective(cls, input1):
        '''
        input1 : List[Integer]

        Expected return type : int
        '''
        # Read only region end
        # Write code here
        pass
```

## Wings 1 DCA - Python - Cross the bridge

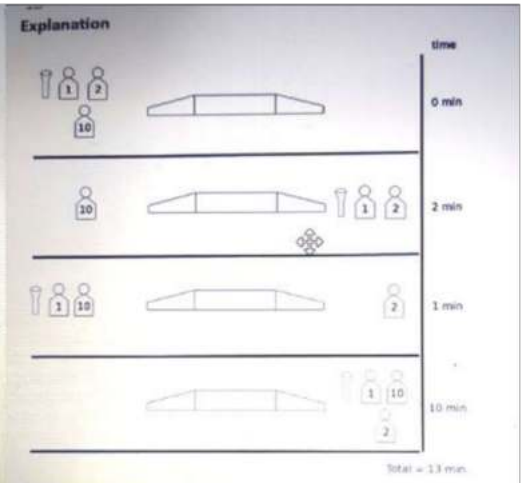
A sea bridge was named a **mysterious bridge**, use this bridge will be opened only at night time. So `n` number of people has to wait till bridge will open. Once it is opened, all of them need to cross the bridge. Since its night time, they need to carry a torch to cross the bridge. The problem is that there is only one torch available, and at most, two people can cross the bridge at a time. When two people cross the bridge together, they walk at a slower person's pace. **Find the minimum time in which each of them can cross the bridge so that all of them cross the bridge with minimum time.**

### For example

There are three people on one side of the bridge. The time taken by each of them is given below.

- Person A takes 1 minute
- Person B takes 2 minutes
- Person C takes 10 minutes

When two people cross the bridge together, they walk at a slower person's pace. If A and C are crossing the bridge together, then it will take them 10




The total time taken to cross the bridge, in this case, is **13 minut**

We will address the person taking 't' time to cross the bridge as person t.

- Initially, we have put the time as zero.
- person-1 and person-2 cross the bridge together, taking the torch with them. Time is **2 minutes** as they move at a slower person's pace.
- person-1 returns with the torch. It takes **1 minute** time.
- person-1 and person-10 cross the bridge together with the torch. It takes **10 minutes**.

The total time taken to cross the bridge, in this case, is **13 minutes**. This is the least possible time to cross the bridge.

**Function Description**

Complete the function ***findMinimumTimeToCross*** in the editor below.
***findMinimumTimeToCross*** has the following parameter:

input1[i]: the time to cross the bridge of each person

**Returns**

int: the minimum total time to cross the bridge for all

**Constraints**

$1 \le n \le 105$

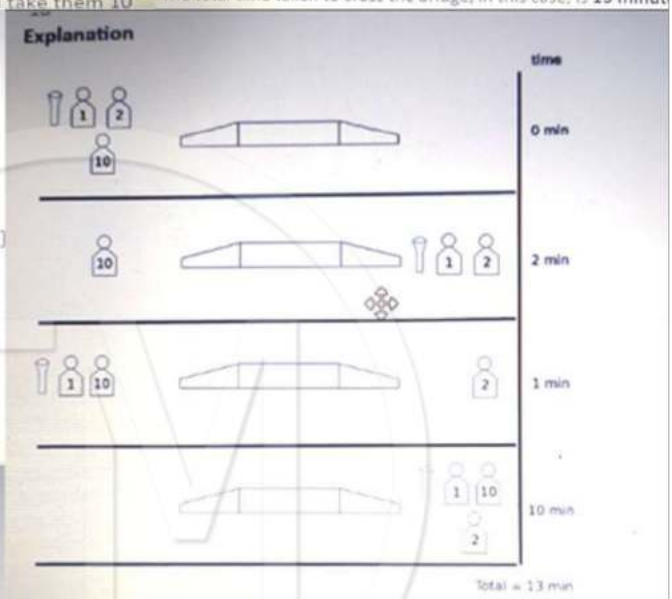
$1 \le input1[i] \le 105$

Let's begin by noting the critical points in this puzzle.

- Torch needed to cross the bridge, and there was only one torch.
- Almost two people can cross the bridge at a time.
- If two people are crossing, then they walk at a slower person's pace.

We can conclude that each time two people cross the bridge taking the torch, one of them returns to bring the torch back.

The first solution that comes to mind is that A is the fastest, so he should be the one to take the torch back each time. Let's see how much time this takes.




The total time taken to cross the bridge, in this case, is **13 minutes**.