Final Exam;
Semester Review;

Computer Science 111
Boston University

Vahid Azadeh-Ranjbar, Ph.D.

CS111 Final Exam

- Final exam: **Thursday, 12/19, 9:00 - 11:00 a.m.**
  - see the final-exam info sheet online
  - students with conflicts have been emailed the alternate exam info
- Bring:
  - A small index card that you obtain from us
  - your BU ID

CS111 Final Exam

- Exam format:
  - Part I: multiple-choice

CS111 Final Exam

- Exam format:
  - Part I: multiple-choice
  - Part II: several multi-part problems
    - some choice (3 out of 4, 4 out of 5, etc.)

CS111 Final Exam

- Exam format:
  - Part I: multiple-choice
  - Part II: several multi-part problems
    - some choice (3 out of 4, 4 out of 5, etc.)
  - Part III: multi-part problem that you must complete
    - parts of this may require a bit more work

CS111 Final Exam

- Exam format:
  - Part I: multiple-choice
  - Part II: several multi-part problems
    - some choice (3 out of 4, 4 out of 5, etc.)
  - Part III: multi-part problem that you must complete
    - parts of this may require a bit more work

- See website for more info, practice problems.

CS111 Final Exam

Part I:

- Tracing functions
- finding the outputs of a program
- Indexing, slicing, skip-slicing
- Mutable and immutable data
- 2D lists
- List comprehension
- Classes, objects & methods
- Dictionary
- Finite State Machine
- Circuits
- HMMM
- Efficiency and classifying algorithms

CS111 Final Exam

Part II and III:

- Object oriented programming
- Making functions:
  - ✓Using loops
  - ✓Using Recursion
  - ✓Using List comprehensions
  - ✓Using helper functions
  - ✓Using any method
- Circuits
- Assembly (HMMM)
- Finite State Machine

Extra Office Hours

I will hold extra office hours to help you for final exam:

- Thursday 12/12 at 3 – 5 PM
- Friday 12/13 at 3 – 5 PM
- Tuesday 12/17 at 2 – 4 PM

If you need to meet me out of office hours, please let me know by email (vranjbar@bu.edu)

TFs and instructors will have office hours during study period.

Some Review Problems

Tracing functions; mutable and immutable data

What is printed by the following program?

```
def mystery(a, b, c, i):
   if i < 2:
      a[i] += 2
   else:
      a = [1, 3]

a = [4, 6]
b = a[:]
c = b

mystery(c, a, b, 2)

print(a, b, c)
```

A. `[4,6] [4,6] [4,6]`
B. `[1,3] [4,6] [4,6]`
C. `[4,6] [1,3] [4,6]`
D. `[1,3] [1,3] [1,3]`
E. none of these

Tracing functions; mutable and immutable data

What is printed by the following program?

```
def mystery(a, b, c, i):
   if i < 2:
      a[i] += 2
   else:
      a = [1, 3]

a = [4, 6]
b = a[:]
c = b

mystery(c, a, b, 2)

print(a, b, c)
```

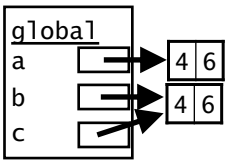
A. `[4,6] [4,6] [4,6]`
B. `[1,3] [4,6] [4,6]`
C. `[4,6] [1,3] [4,6]`
D. `[1,3] [1,3] [1,3]`
E. none of these

Tracing functions; mutable and immutable data

What is printed by the following program?

```
def mystery(a, b, c, i):
    if i < 2:
        a[i] += 2
    else:
        a = [1, 3]
a = [4, 6]
b = a[:]
c = b
mystery(c, a, b, 2)
print(a, b, c)
```

before `mystery` *during* `mystery` *after* `mystery`

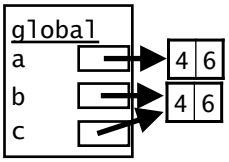


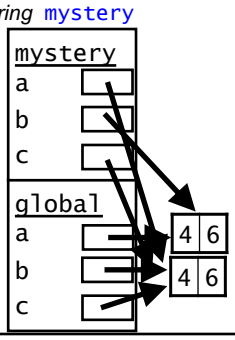
Tracing functions; mutable and immutable data

What is printed by the following program?

```
def mystery(a, b, c, i):
    if i < 2:
        a[i] += 2
    else:
        a = [1, 3]
a = [4, 6]
b = a[:]
c = b
mystery(c, a, b, 2)
print(a, b, c)
```

before `mystery` *during* `mystery` *after* `mystery`



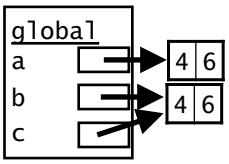


Tracing functions; mutable and immutable data

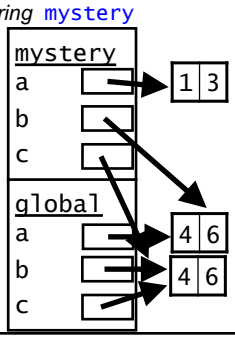
What is printed by the following program?

```
def mystery(a, b, c, i):
    if i < 2:
        a[i] += 2
    else:
        a = [1, 3]
a = [4, 6]
b = a[:]
c = b
mystery(c, a, b, 2)
print(a, b, c)
```

before `mystery`



during `mystery`



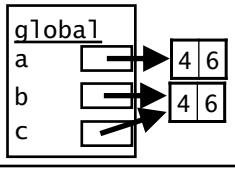
after `mystery`

Tracing functions; mutable and immutable data

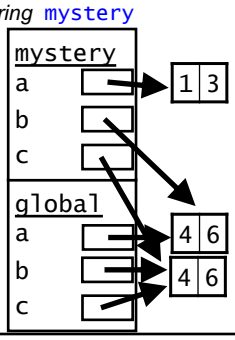
What is printed by the following program?

```
def mystery(a, b, c, i):
    if i < 2:
        a[i] += 2
    else:
        a = [1, 3]
a = [4, 6]
b = a[:]
c = b
mystery(c, a, b, 2)
print(a, b, c)
```

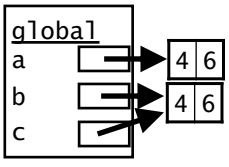
before `mystery`



during `mystery`



after `mystery`



Dictionary
What Is the Output?

```
d = {5: 18, 7: 12, 9: 21}

count = 0
for x in d:
    if x > 15:
        count = d[x]

print(count)
```

A. 18
B. 9
C. 21
D. 0
E. none of these

Dictionary
What Is the Output?

```
d = {5: 18, 7: 12, 9: 21}

count = 0
for x in d: # x gets one key at a time!
    if x > 15:
        count = d[x]

print(count)
```

A. 18
B. 9
C. 21
D. 0
E. none of these

Dictionary
What Is the Output?

```
d = {5: 18, 7: 12, 9: 21}

count = 0
for x in d: # x equals 5 then 7 then 9
    if x > 15:
        count = d[x]

print(count)
```

A. 18

B. 9
C. 21

D. 0

E. none of these

Dictionary
What Is the Output?

```
d = {5: 18, 7: 12, 9: 21}

count = 0
for x in d: # 5, 7 and 9 are less than 15
    if x > 15:
        count = d[x]

print(count) # initial value of count = 0 will be printed.
```

A. 18

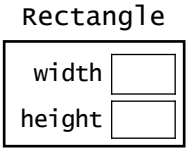
B. 9
C. 21

D. 0

E. none of these

What is the output of this program?

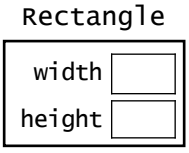
```
from rectangle import *
                    width height
r1 = Rectangle(20, 30)
r2 = Rectangle(25, 20)
r3 = r2
r2 = r1
r1 = r3

print(r1.width, r2.width, r3.width)
```



A. 20 20 20
B. 25 25 25
C. 25 20 25
D. 20 25 20
E. 25 25 20
F. none of these

What is the output of this program?

```
from rectangle import *
                    width height
r1 = Rectangle(20, 30)
r2 = Rectangle(25, 20)
r3 = r2
r2 = r1
r1 = r3

print(r1.width, r2.width, r3.width)
```

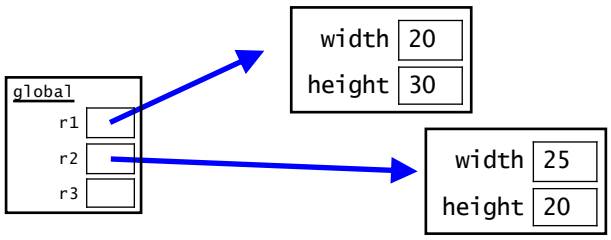


A. 20 20 20
B. 25 25 25
C. 25 20 25
D. 20 25 20
E. 25 25 20
F. none of these

What is the output of this program?

Rectangle

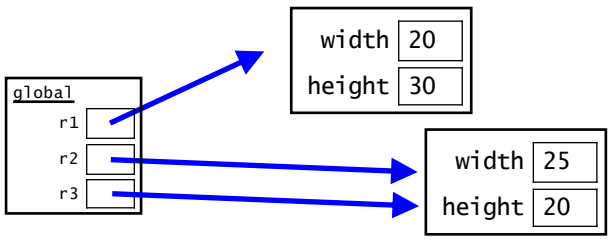
```
from rectangle import *
                    width height
r1 = Rectangle(20, 30)
r2 = Rectangle(25, 20)
r3 = r2
r2 = r1
r1 = r3

print(r1.width, r2.width, r3.width)
```



What is the output of this program?

Rectangle

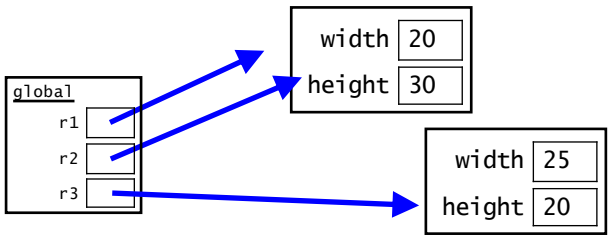
```
from rectangle import *
                    width height
r1 = Rectangle(20, 30)
r2 = Rectangle(25, 20)
r3 = r2
r2 = r1
r1 = r3

print(r1.width, r2.width, r3.width)
```



What is the output of this program?

Rectangle

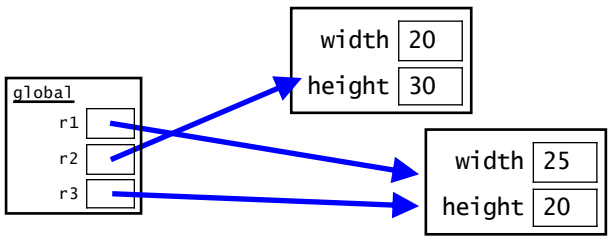
```
from rectangle import *
                    width height
r1 = Rectangle(20, 30)
r2 = Rectangle(25, 20)
r3 = r2
r2 = r1
r1 = r3

print(r1.width, r2.width, r3.width)
```



What is the output of this program?

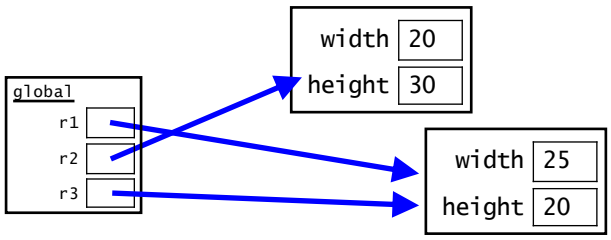
Rectangle

```
from rectangle import *
                    width height
r1 = Rectangle(20, 30)
r2 = Rectangle(25, 20)
r3 = r2
r2 = r1
r1 = r3

print(r1.width, r2.width, r3.width)
```



What is the output of this program?

```
from rectangle import *                    Rectangle
                          width height
r1 = Rectangle(20, 30)
r2 = Rectangle(25, 20)
r3 = r2
r2 = r1
r1 = r3

print(r1.width, r2.width, r3.width) 25 20 25
```



For the Inputs At Right, What Is the Output of this Program?

Screen
6 (1st input)
8 (2nd input)

A. 6
B. 8
C. 12
D. 14
E. none of these

```
0  read r1
1  read r2
2  copy r3 r1
3  sub r4 r2 r1
4  jeqz r4 12
5  nop
6  nop
7  nop
8  nop
9  addn r1 2
10 add r3 r3 r1
11 jumpn 3
12 write r3
13 halt
```

For the Inputs At Right, What Is the Output of this Program?

Screen
6 (1st input)
8 (2nd input)

A. 6
B. 8
C. 12
D. 14
E. none of these

```
0  read r1
1  read r2
2  copy r3 r1
3  sub r4 r2 r1
4  jeqz r4 12
5  nop
6  nop
7  nop
8  nop
9  addn r1 2
10 add r3 r3 r1
11 jumpn 3
12 write r3
13 halt
```

For the Inputs At Right, What Is the Output of this Program?

Screen
6 (1st input)
8 (2nd input)

r1	6
r2	8
r3	6
r4	2

```
0  read r1
1  read r2
2  copy r3 r1
3  sub r4 r2 r1
4  jeqz r4 12
5  nop
6  nop
7  nop
8  nop
9  addn r1 2
10 add r3 r3 r1
11 jumpn 3
12 write r3
13 halt
```

For the Inputs At Right, What Is the Output of this Program?

Screen
6 (1st input)
8 (2nd input)

r1	8
r2	8
r3	6
r4	2

```
0  read r1
1  read r2
2  copy r3 r1
3  sub r4 r2 r1
4  jeqz r4 12
5  nop
6  nop
7  nop
8  nop
9  addn r1 2
10 add r3 r3 r1
11 jumpn 3
12 write r3
13 halt
```

For the Inputs At Right, What Is the Output of this Program?

Screen
6 (1st input)
8 (2nd input)

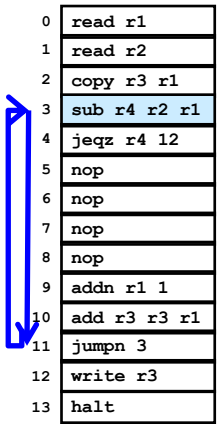
r1	8
r2	8
r3	14
r4	2

```
0  read r1
1  read r2
2  copy r3 r1
3  sub r4 r2 r1
4  jeqz r4 12
5  nop
6  nop
7  nop
8  nop
9  addn r1 2
10 add r3 r3 r1
11 jumpn 3
12 write r3
13 halt
```

For the Inputs At Right,
What Is the Output of this Program?

Screen
6 (1st input)
10 (2nd input)

r1	8
r2	8
r3	14
r4	2

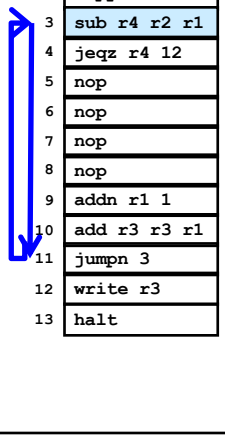


```
0  read r1
1  read r2
2  copy r3 r1
3  sub r4 r2 r1
4  jeqz r4 12
5  nop
6  nop
7  nop
8  nop
9  addn r1 1
10 add r3 r3 r1
11 jumpn 3
12 write r3
13 halt
```

For the Inputs At Right,
What Is the Output of this Program?

Screen
6 (1st input)
10 (2nd input)

r1	8
r2	8
r3	14
r4	0

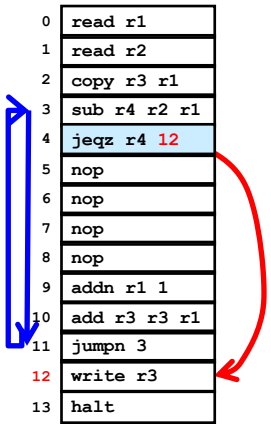


```
0  read r1
1  read r2
2  copy r3 r1
3  sub r4 r2 r1
4  jeqz r4 12
5  nop
6  nop
7  nop
8  nop
9  addn r1 1
10 add r3 r3 r1
11 jumpn 3
12 write r3
13 halt
```

For the Inputs At Right, What Is the Output of this Program?

Screen
6 (1st input)
10 (2nd input)

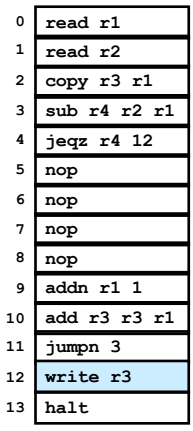
r1	8
r2	8
r3	14
r4	0



For the Inputs At Right, What Is the Output of this Program?

Screen
6 (1st input)
10 (2nd input)
14 (output)

r1	8
r2	8
r3	14
r4	0



How Would You Complete This Function?

```
def longest_word(words):
    """ returns the string that is the longest
        word from the input list of words
    """
    scored_words = ____________________________
    bestpair = max(scored_words)
    return ________________
```

	first blank	second blank
A.	`[[w, len(w)] for w in words]`	`bestpair[0]`
B.	`[[len(w), w] for w in words]`	`bestpair[0]`
C.	`[[w, len(w)] for w in words]`	`bestpair[1]`
D.	`[[len(w), w] for w in words]`	`bestpair[1]`
E.	more than one of these would work	

How Would You Complete This Function?

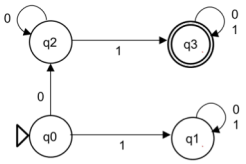
```
def longest_word(words):
    """ returns the string that is the longest
        word from the input list of words
    """
    scored_words = ____________________________
    bestpair = max(scored_words)
    return ________________
```

	first blank	second blank
A.	`[[w, len(w)] for w in words]`	`bestpair[0]`
B.	`[[len(w), w] for w in words]`	`bestpair[0]`
C.	`[[w, len(w)] for w in words]`	`bestpair[1]`
D.	**`[[len(w), w] for w in words]`**	**`bestpair[1]`**
E.	more than one of these would work	

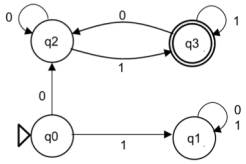
Which of these is the correct FSM?

- Construct a FSM accepting bit strings in which:
  - the **first** bit is 0
  - the **last** bit is 1

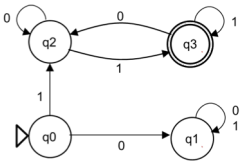
A.



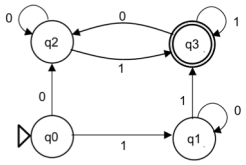
C.



B.



D.

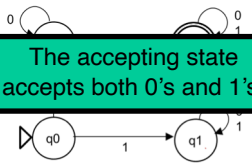


Which of these is the correct FSM?

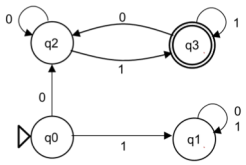
- Construct a FSM accepting bit strings in which:
  - the **first** bit is 0
  - the **last** bit is 1

A.

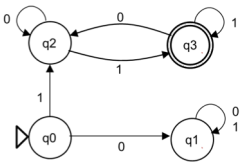
The accepting state accepts both 0's and 1's!



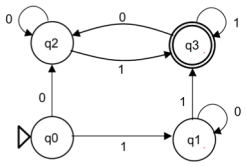
C.



B.



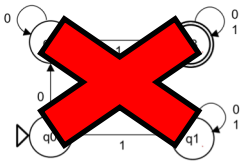
D.



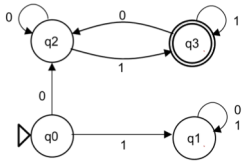
Which of these is the correct FSM?

- Construct a FSM accepting bit strings in which:
  - the **first** bit is 0
  - the **last** bit is 1

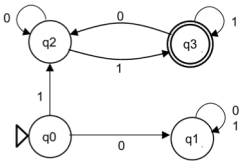
A.



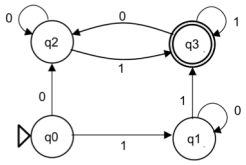
C.



B.



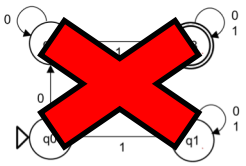
D.



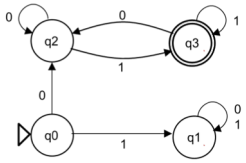
Which of these is the correct FSM?

- Construct a FSM accepting bit strings in which:
  - the **first** bit is 0
  - the **last** bit is 1

A.

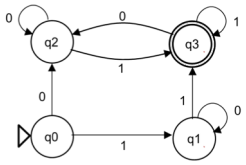


C.



If first input is a zero it transitions into a non accepting state and stays there!

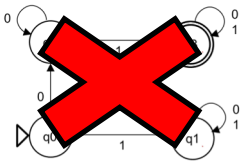
D.



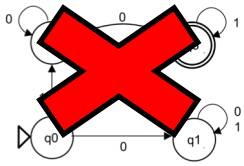
Which of these is the correct FSM?

- Construct a FSM accepting bit strings in which:
  - the **first** bit is 0
  - the **last** bit is 1

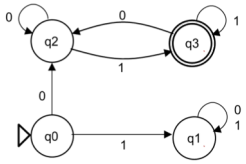
A.



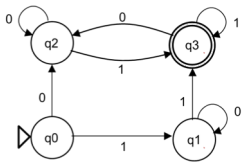
B.



C.



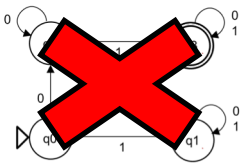
D.



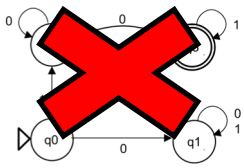
Which of these is the correct FSM?

- Construct a FSM accepting bit strings in which:
  - the **first** bit is 0
  - the **last** bit is 1

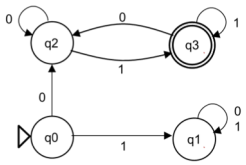
A.



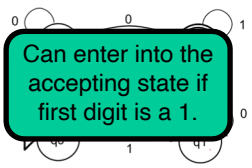
B.



C.



D.

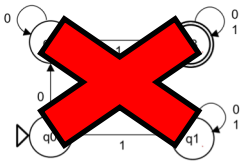


Can enter into the accepting state if first digit is a 1.

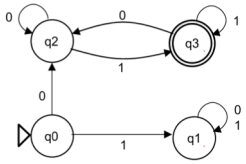
Which of these is the correct FSM?

- Construct a FSM accepting bit strings in which:
  - the **first** bit is 0
  - the **last** bit is 1

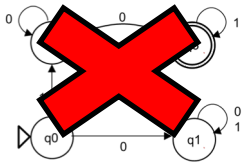
A.



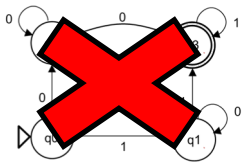
C.



B.



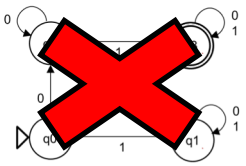
D.



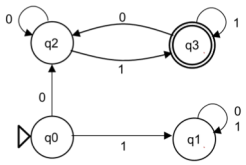
Which of these is the correct FSM?

- Construct a FSM accepting bit strings in which:
  - the **first** bit is 0
  - the **last** bit is 1

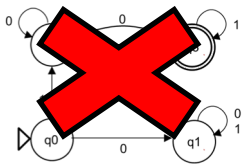
A.



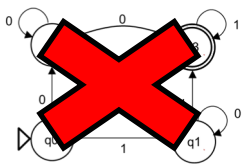
C.



B.



D.



Algorithm Analysis

- Computer scientists characterize an algorithm's efficiency by specifying its *growth function*.
  - the function to which its running time is roughly proportional
- We've seen several different growth functions:

```
_____    # binary search
_____    # sequential/linear search
_____    # quicksort
_____    # selection sort
```

- Others include:

```
_____    # exponential growth
_____    # factorial growth
```

Algorithm Analysis

- Computer scientists characterize an algorithm's efficiency by specifying its *growth function*.
  - the function to which its running time is roughly proportional
- We've seen several different growth functions:

```
log2n    # binary search
n        # sequential/linear search
n log2n  # quicksort
n2       # selection sort
```

- Others include:

```
cn       # exponential growth
n!       # factorial growth
```

- There are a 2 bits number (xy) and a 1 bit number (z).
- We want to add these numbers and return True if the addition is Even.
- Complete the Truth table and select the correct minterm expansion.

inputs			output
x	y	z	
0	0	0	1

A. $yz + xz + xy$

B. $xyz + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z$

C. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$

D. $\bar{x}\bar{y}\bar{z} + \bar{x}yz + x\bar{y}\bar{z} + xy\bar{z}$

E. none of the above

- There are a 2 bits number (xy) and a 1 bit number (z).
- We want to add these numbers and return True if the addition is Even.
- Complete the Truth table and select the correct minterm expansion.

inputs			output
x	y	z	
0	0	0	1 (integer 0)
0	0	1	0 (integer 1)

A. $yz + xz + xy$

B. $xyz + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z$

C. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$

D. $\bar{x}\bar{y}\bar{z} + \bar{x}yz + x\bar{y}\bar{z} + xy\bar{z}$

E. none of the above

- There are a 2 bits number (xy) and a 1 bit number (z).
- We want to add these numbers and return True if the addition is Even.
- Complete the Truth table and select the correct minterm expansion.

inputs			output
x	y	z	
0	0	0	1 (integer 0)
0	0	1	0 (integer 1)
0	1	0	0 (integer 1)

A. $yz + xz + xy$

B. $xyz + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z$

C. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$

D. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xy\bar{z}$

E. none of the above

- There are a 2 bits number (xy) and a 1 bit number (z).
- We want to add these numbers and return True if the addition is Even.
- Complete the Truth table and select the correct minterm expansion.

inputs			output
x	y	z	
0	0	0	1 (integer 0)
0	0	1	0 (integer 1)
0	1	0	0 (integer 1)
0	1	1	1 (integer 2)
1	0	0	1 (integer 2)
1	0	1	0 (integer 3)
1	1	0	0 (integer 3)
1	1	1	1 (integer 4)

A. $yz + xz + xy$

B. $xyz + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z$

C. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$

D. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xy\bar{z}$

E. none of the above

- There are a 2 bits number (xy) and a 1 bit number (z).
- We want to add these numbers and return True if the addition is Even.
- Complete the Truth table and select the correct minterm expansion.

inputs			output
x	y	z	
0	0	0	1 (integer 0)
0	1	1	1 (integer 2)
1	0	0	1 (integer 2)
1	1	1	1 (integer 4)

A. $yz + xz + xy$

B. $xyz + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z$

C. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$

D. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xy\bar{z}$

E. none of the above

- There are a 2 bits number (xy) and a 1 bit number (z).
- We want to add these numbers and return True if the addition is Even.
- Complete the Truth table and select the correct minterm expansion.

inputs			output
x	y	z	
0	0	0	1 (integer 0)
0	1	1	1 (integer 2)
1	0	0	1 (integer 2)
1	1	1	1 (integer 4)

A. $yz + xz + xy$

B. $xyz + x\bar{y}\bar{z} + \bar{x}y\bar{z} + \bar{x}\bar{y}z$

C. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xyz$

D. $\bar{x}\bar{y}\bar{z} + \bar{x}y\bar{z} + x\bar{y}\bar{z} + xy\bar{z}$

E. none of the above

$\bar{x}\bar{y}\bar{z} + \bar{x}yz + x\bar{y}\bar{z} + xyz$