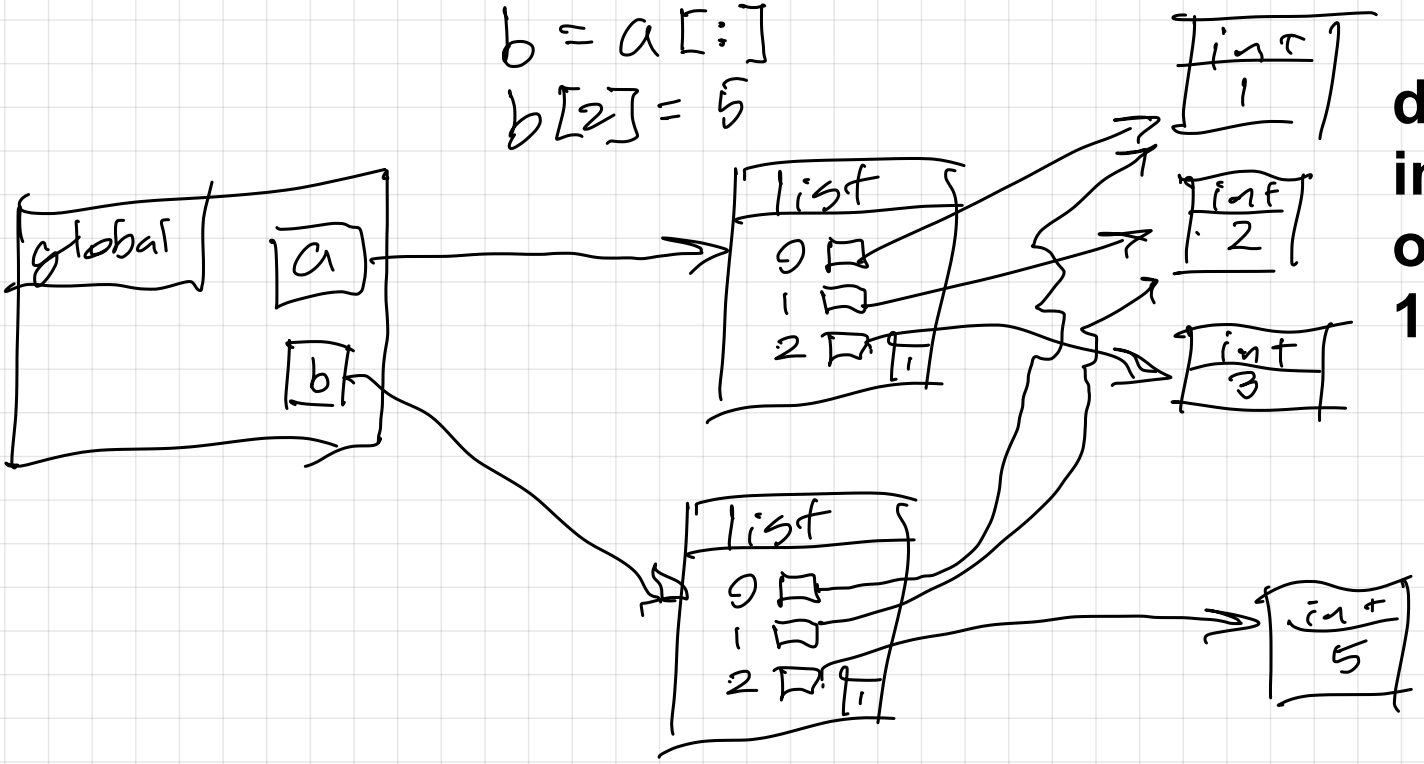
Aliasing: När två eller fler variabelnamn pekar på samma objekt.

---

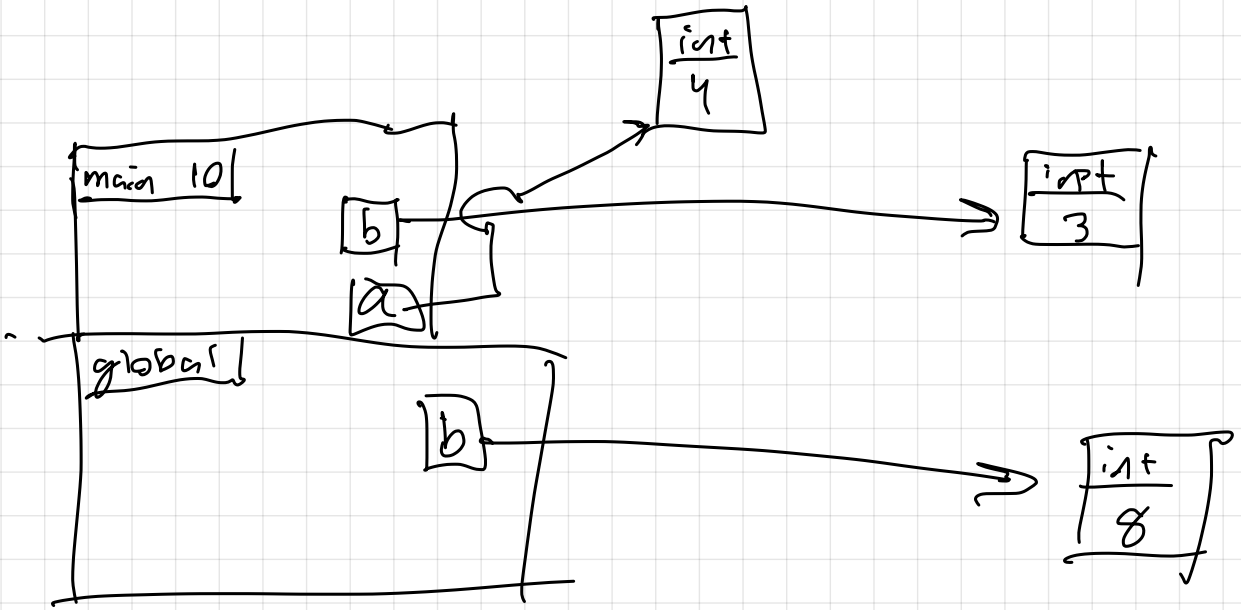
```
a = [1,2,3]
b = a[:]
b[2] = 5
```



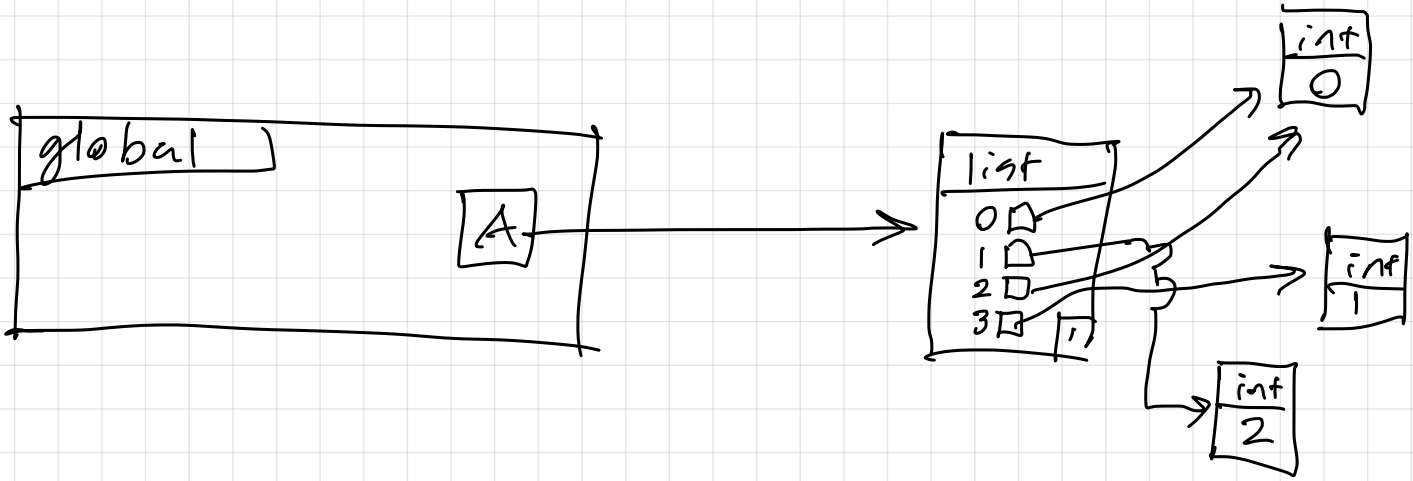

**detta är copying inte aliasing, info om aliasing på sida 106**

Rita tid- och-pil diagrammet för hur minnet ser ut vid den kommenterade raden.

```
1  def f(n):
2      b = n+1
3      return b

4  def main():
5      b = 3
6      a = f(b)
7      #komment.
8
9  b = 8
10 main()
```



```
1 def main():
2       b= A
3       b[1] =2
4
5 A = [0,0,0,1]
6 main()
7 #komment
```




a = [0, 0]

b = a

b[0] = 1

print(a) ⟶ [1, 0]

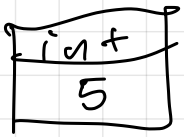
id(a) == id(b) True: samma objekt

False: olika objekt

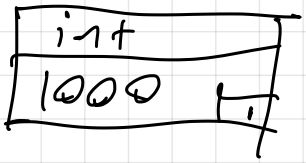
index: Behövs för objekt som inte ALLTID finns i minnet

Vad finns alltid i minnet?

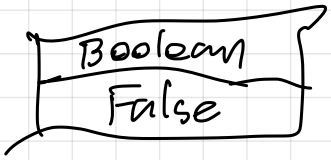
| Objekt | värden |
| --- | --- |
| int | [-5, 255] |
| Boolean | True, False |
| NoneType | None |

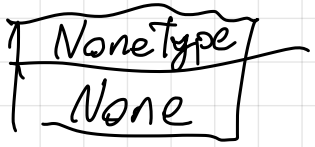


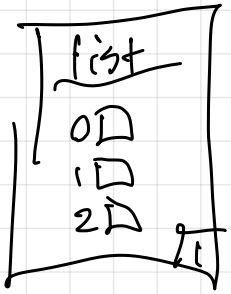


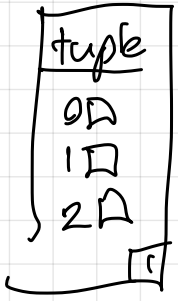


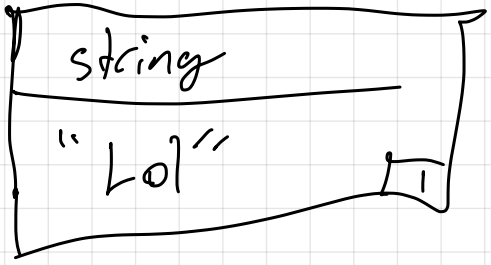


(16) Byt ut kommentaren i koden nedan mot ett svarsalternativ i taget. Vilket/vilka alternativ får detta Python3-program att exekvera raden med print?

```
def main():
    # Här ska svaret stoppas in i koden.
    if ( not a or b) and (not b or c):
        print(a,b,c)


main()
```

```
(A) a, b, c = True, True, True
(B) a, b, c = True, True, False
(C) a, b, c = True, False, True
(D) a, b, c = True, False, False
(E) a, b, c = False, True, True
(F) a, b, c = False, True, False
(G) a, b, c = False, False, True
(H) a, b, c = False, False, False
(I) Inget av ovanstående alternativ.
```

not a or b :
≠
(not a) or b :

Antingen a = False eller b = True ger True

(not b) or c : Antingen b = False c = True

(A) True (B) False (C) False (D) False

(E) True (F) False (G) True (H) True

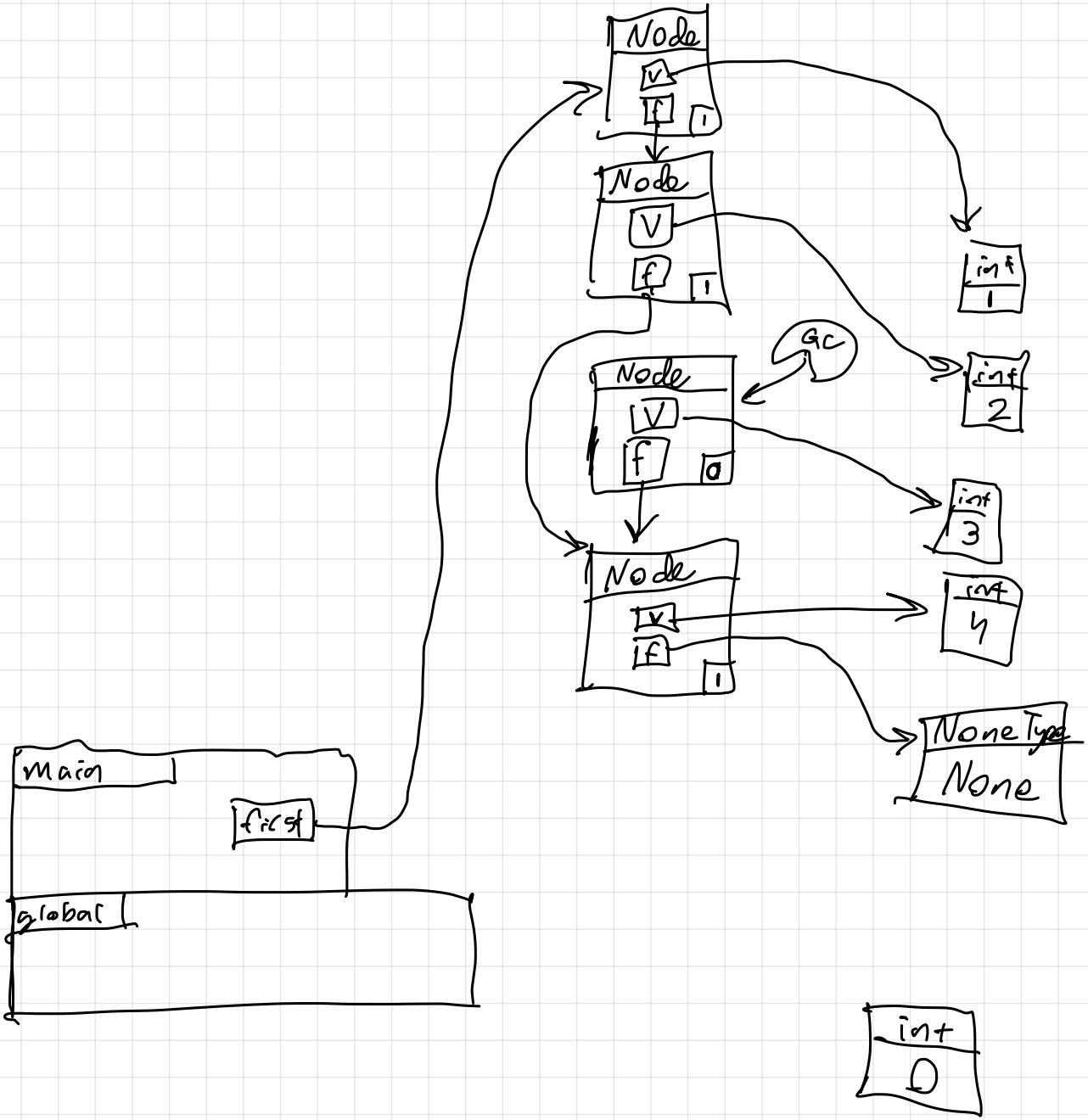
Svar: A, E, G, H

```
first = destroy_element(first, 2)
```

...i programmet nedan:

```
class Node:
    def __init__(self, value, following):
        self.value = value
        self.following = following


def print_values(first):
    temp = first
    while temp:
        print(temp.value)
        temp = temp.following


def create_list(size):
    first = Node(size, None)
    for i in range(size-1, 0, -1):
        first = Node(i, first)
    return first


def destroy_element(first, index):
    """returns the first element where index has been destroyed"""
    if index == 0:
        return first.following
    answer = first #we will change this variable
    for i in range(index-1):
        first = first.following
    following_following = first.following.following
    first.following = following_following
    return answer


def main():
    first = create_list(4)
    print_values(first)
    first = destroy_element(first, 2)
    print_values(first)


if __name__ == '__main__':
    main()
```

efter denna funktion är klar



(33) Vad skriver följande program ut?

```python
def mystery(n):
    if n <= 0:
        return 1
    return mystery(n-1)+mystery(n-2)


print(mystery(3))
```

1. 1
2. 2
3. 3
4. 4
5. 5
6. 6
7. 7
8. 8
9. 9
10. Inget av ovanstående alternativ

$m(3) = m(2) + m(1) = 5$

$m(2) = m(1) + m(0) = 3$

$m(1) = m(0) + m(-1) = 2$

$m(0) = 1$

$m(-1) = 1$