(1)

```c
int x;
int y;
int z;
int* w;
int* q;
x = 0;
y = 1;
z = 2;
w = &x;
q = &y;
*w = y;
*q = z;
*w = x + y + z + *q;
*q = x + y + z + *w;
printf("x=%d, y=%d, z=%d",x,y,z);
```

x	w = &x *w (x) = y (x=1) *w (x) = x + y + z + *q = 1 + 2 + 2 + 2 = **<u>7</u>**
y	q = &y *q (y) = z (y=2) *q (y) = x + y + z + w* = 7 + 2 + 2 + 7 = **<u>18</u>**
z	= **<u>2</u>**

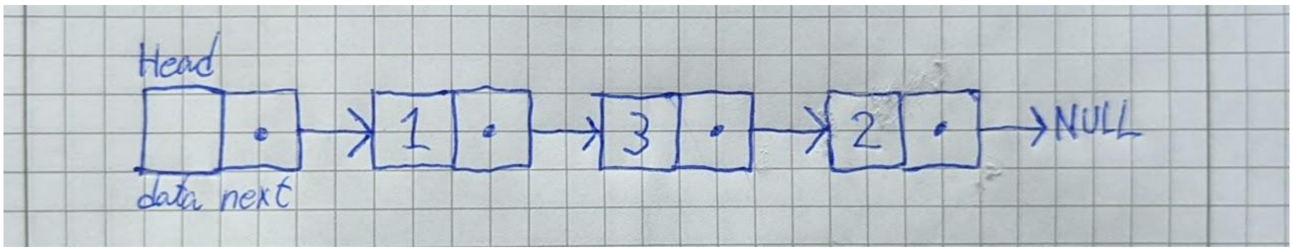
(2)

```c
int max(int* numbers, int size) {
    assert(size>0); //Precondition: array not empty
    int max = *numbers; //assign 'max' first element of array, pointed to by *numbers
    for (int i = 0; i < size; i++, numbers++) { //with each iteration pointer '*number' is
                                                //incremented so to point at the next element
        (*numbers > max) && (max = *numbers); //if value at *number is > max, its becomes new max
    }
    return max;
}
```

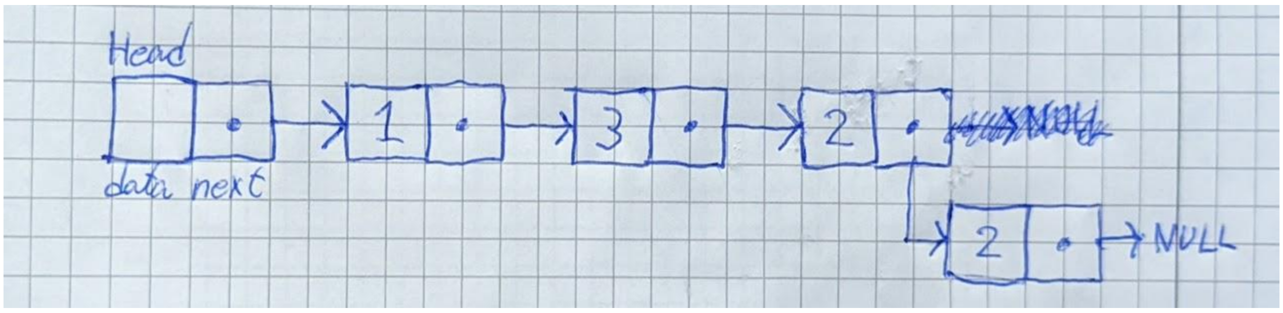
(3)

a.

Ved linje 31



Ved linje 33



b.

```
int size(node *l){
    assert(l!=NULL); //Precondition
    int size = 0;
    while (l->next != NULL) { //loops through array until 'next' value is NULL
        size++;  //increment size with each iteration
        l = l->next; //the current node of list becomes the next node
    }
    return size;
}
```

c.

node *p = l->next;	p points to next node in the list
while (p!=NULL){	Loops while the current node isn't Empty (NULL) Current node is never changes, so loop will never end. Unless initial node is already NULL.
printf("\%d, ",p->data);	Prints the data from current Node
printf("\n");	Prints on a new line (never reached)

The code will continuously print the same element from the list, and so does not fulfil the postcondition.

d.

```c
void printout(node *l) {
    node *p = l->next; //skips first empty element
    while (p != NULL) { //loops until empty element is found
        printf("\%d, ",p->data); //prints data
        p = p->next; //points to the next element in list
    }
    printf("\n");
}
```

e.

```c
int largest(node *l) {
    int max = l->next->data; //skips first empty element -> gets data next
    while (l->next != NULL) { //loops until empty element is found
        l = l -> next; //points to the next element in list
        (l->data > max) && (max = l->data); //if value at l->data is > max, its becomes new max
    }
    return max;
}
```