Riešenie testu

1. Máme daný slovník `d = {'bum' :7, 'bom' :3, 'bam' :5, 'bim' :6, 'bem' :2}`
   Zistite, čo vráti tento kód:

```
'.'.join(b for a, b in sorted((d[x],x) for x in d))
```

```
'bem.bom.bam.bim.bum'
```

bodovanie: **3 body**

2. Vrchol spájaného zoznamu je definovaný takto:

```
class Vrchol:
    def __init__(self, data, next=None):
        self.data, self.next = data, next
```

Napíšte funkciu **kopia**, ktorá dostáva ako parameter začiatok nejakého zoznamu a vráti nový zoznam, ktorý je kópiou pôvodného.

bodovanie: **8-10 bodov**

```
def kopia(zoznam):
    if zoznam is None:
        return None
    vysl = p = Vrchol(zoznam.data)
    while zoznam.next is not None:
        zoznam = zoznam.next
        p.next = Vrchol(zoznam.data)
        p = p.next
    return vysl
```

alebo to isté rekurzívne:

```
def kopia(zoznam):
    if zoznam is None:
        return None
    return Vrchol(zoznam.data, kopia(zoznam.next))
```

3. Do triedy **Zoznam** dopíšte metódu **vyhod2**, ktorá z momentálneho zoznamu vyhodí každý druhý vrchol.

```
class Vrchol:
    def __init__(self, data, next):
        self.data = data
        self.next = next
class Zoznam:
    def __init__(self):
        self.zac = None
    def vyhod2(self):
        pom = self.zac
        while pom is not None and pom.next is not None:
            pom.next = pom.next.next
            pom = pom.next
```

bodovanie: **5 bodov**
1.časť podmienky: 1bod
2.časť podnienky: 2 body
priradenie: 2 body

4. Na začiatku bol ***zásobník*** prázdny. Potom sme vykonali niekoľko operácií **push** a **pop**.
   Zistite, čo sa vypíše po spustení:

```
print('pre zasobnik:')
for x in 7,3,5,11,3,8,4: push(x)
for x in range(6):
    print(pop(), pop(), end=' ')
    push(x)
print(pop())
```

```
pre zasobnik:
4 8 0 3 1 11 2 5 3 3 4 7 5
```

bodovanie: **6 bodov**
zásobník: 3 body
rad: 3 body

Na začiatku bol ***rad*** prázdny. Potom sme vykonali niekoľko operácií **enqueue** a **dequeue**.
Zistite, čo sa vypíše po spustení:

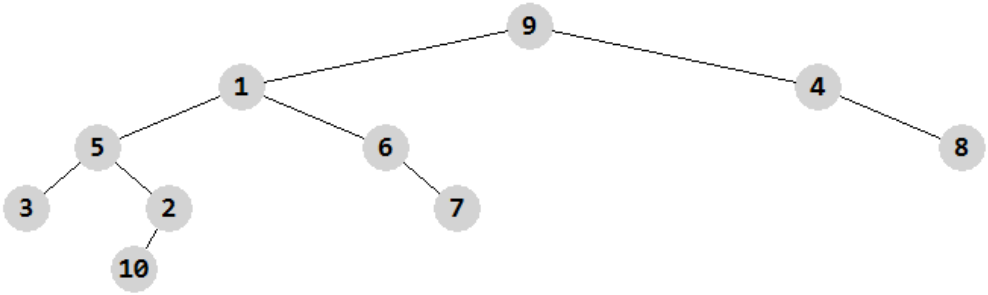
```
print('pre rad:')
for x in 7,3,5,11,3,8,4: enqueue(x)
for x in range(6):
    print(dequeue(), dequeue(), end=' ')
    enqueue(x)
print(dequeue())
```

```
pre rad:
7 3 5 11 3 8 4 0 1 2 3 4 5
```

5. Máme dané dva výpisy toho istého binárneho stromu. Nakreslite tvar tohto stromu a vypíšte aj **strom.preorder()**

```
>>> print(strom.inorder())
3 5 10 2 1 6 7 9 4 8
>>> print(strom.postorder())
3 10 2 5 7 6 1 8 4 9
```

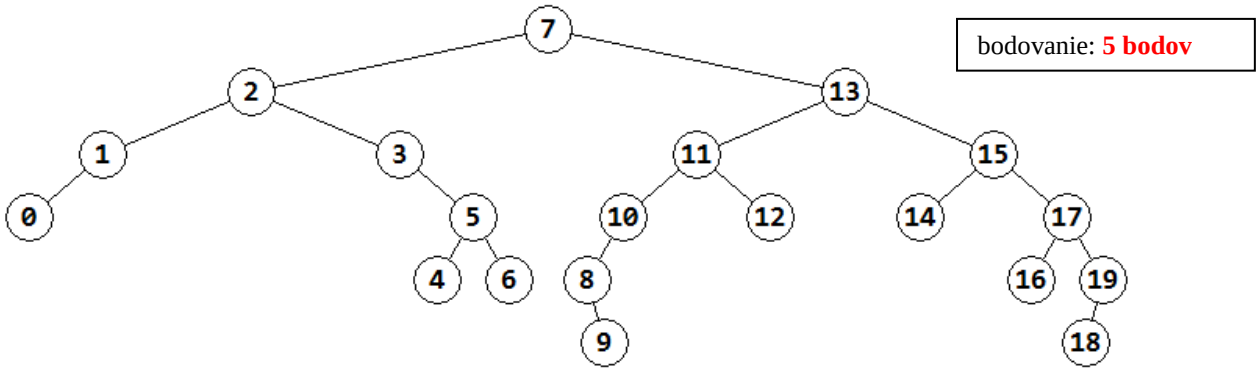
bodovanie: **6 bodov**




```
preorder = 9 1 5 3 2 10 6 7 4 8
```

6. Dopíšte do stromu na obrázku čísla z postupnosti **range(20)** tak, aby vznikol binárny vyhľadávací strom:

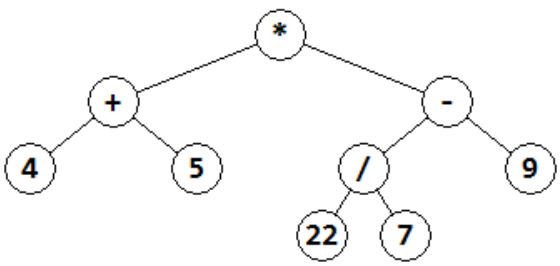
bodovanie: **5 bodov**



7. Nakreslite aritmetický strom, pre ktorý je táto postupnosť **postorderom**:

```
4 5 + 22 7 / 9 - *
```

bodovanie: **4 body**



8. Máme daný algoritmus **quick sort**:

```
def quick_sort(pole):
    def rozdel(z,k):
        pivot, index = pole[z], z
        for i in range(z+1, k+1):
            if pole[i] < pivot:
                index += 1
                pole[i], pole[index] = pole[index], pole[i]
        pole[z], pole[index] = pole[index], pole[z]
        return index
    def quick_sort1(z, k):
        if z < k:
            index = rozdel(z,k)
            quick_sort1(z, index-1)
            quick_sort1(index+1, k)
    quick_sort1(0, len(pole)-1)
```

bodovanie: **4 body**

Zapíšte obsah poľa po každom návrate z funkcie **rozdel**, pričom na začiatku boli v poli:

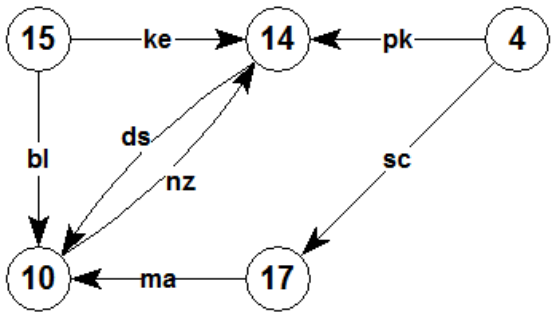
68	**58**	**25**	**32**	**88**	**82**	**37**	**86**	**95**	**15**
15	58	25	32	37	68	88	86	95	82
15	58	25	32	37					
	37	25	32	58					
	32	25	37						
	25	32							
						82	86	88	95
						82	86		

Prvok v riadku tabuľky, ktorý bol pritom pivotom, zakrúžkujte.

9. Nakreslite **orientovaný ohodnotený graf**, ktorý sme zapísali pomocou asociatívneho poľa vrcholov s asociatívnymi poľami susedov:

```
graf = {15:{14:'ke',10:'bl'},4:{17:'sc',14:'pk'},10:{14:'nz'},
        17:{10:'ma'},14:{10:'ds'}}
```

bodovanie: **5 bodov**



10. Napíšte program pre **Turingov stroj**, ktorý bude akceptovať vstup na páske len vtedy, keď je na nej číslo v dvojkovej sústave, pričom toto číslo je nepárne. Koncový stav má meno `'end'`. Hlava je nad začiatkom vstupu:

bodovanie: **4 body**

```
prog = '''
0 0 0 > 0
0 1 1 > 0
0 _ _ < 2
2 1 1 = end
'''     # kde pravidla su v tvare: stav1 znak1 znak2 posun stav2
for vstup in '', '11a', '0101010101', '11110', '1', '0':
    t = Turing(prog, vstup)
    print(repr(vstup), t.rob(vypis=False))
```