

Telekommunikációs Hálózatok

2. gyakorlat

PYTHON ALAPOK II.

Fájl átvitel

- fájl bináris megnyitása

```
with open („input.txt”, „rb”) as f:  
    ...
```

- read(x)
 - x byte beolvasása (ha binárisra van megnyitva)
 - x karakter beolvasása (ha file olvasásra van megnyitva)

```
    ...  
    f.read(128) #128 byte-ot fog beolvasni
```

„When size is omitted or negative, the entire contents of the file will be read and returned; it’s your problem if the file is twice as large as your machine’s memory. „ - python.org

Struktúráküldése

- Binárisra alakítjuk az adatot

```
import struct
values = (1, "ab".encode(), 2.7)
packer = struct.Struct("i 2s f")           #Int, char[2], float
packed_data = packer.pack(*values)
```

- Visszalakítjuk a kapott üzenetet

```
import struct
unpacker = struct.Struct("i 2s f")
unpacked_data = unpacker.unpack(data)
```

- megj.: integer 1 – 4 byte, stringként 1 byte, azaz hatékonyabb stringként átküldeni.

Struktúra jellemzői

- Mire kell figyelni?
 - A Struct formátumnál az "Xs" (pl. "2s") X db. bájtból álló **bájtliterált** jelent (pl. **b'abc'**)

```
import struct
values = (1, 'ab', 2.7)
packer = struct.Struct('i 2s f')
packed_data = packer.pack(*values)
# HIBA: struct.error: argument for 's' must be a bytes object
# JÓ megoldás:
values = (1, b'ab', 2.7) # vagy values = (1, 'ab'.encode(), 2.7)
...
```

Struktúra jellemzői

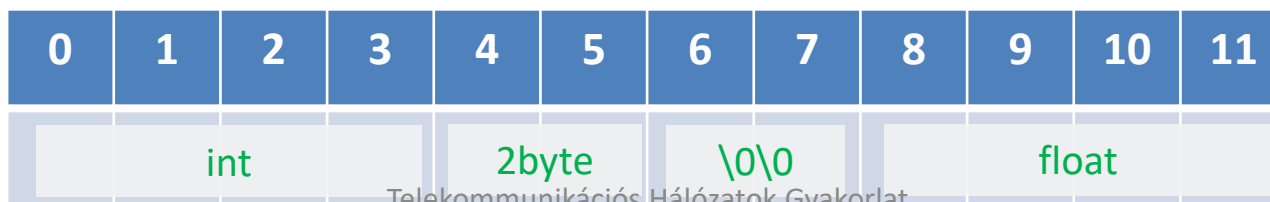
- Mire kell figyelni?

- A struktúra mérete byte-ban:

```
import struct
packer = struct.Struct("i 2s f")
print(struct.calcsize("i 2s f "))
print(packer.size)
# 12
```

- i: int size = 4, 2s: 2 bytes, f: float size = 4,
4+2+4 ≠ **12** ???

- Az int/float-ot úgy igazítja, hogy a kezdő pozíciója 4-gyel osztható legyen



Struktúra

Character	Byte order
@	native
=	native
<	little-endian
>	big-endian
!	network (= big-endian)

Format	C Type	Python type	Standard size
x	pad byte	no value	
c	char	bytes of length 1	1
b	signed char	integer	1
B	unsigned char	integer	1
?	_Bool	bool	1
h	short	integer	2
H	unsigned short	integer	2
i	int	integer	4
I	unsigned int	integer	4
l	long	integer	4
L	unsigned long	integer	4
q	long long	integer	8
Q	unsigned long long	integer	8
n	ssize_t	integer	
N	size_t	integer	
e	(f)	float	2
f	float	float	4
d	double	float	8
s	char[]	bytes	
p	char[]	bytes	
P	void*	integer	

Fájlkezelés

- **SEEK:** a fájlobjektumnak az aktuális pozíciója:

```
with open('alma.txt', 'r') as f:
    sor = f.readline()
    print('jelenlegi sor', sor)           # jelenlegi sor 1. sor

    sor = f.readline()
    print('jelenlegi sor', sor)           # jelenlegi sor 2. sor

    f.seek(0, 0)                          # f.seek(offset, whence)

    sor = f.readline()
    print('jelenlegi sor', sor)           # jelenlegi sor 1. sor
```

- offset: olvasás/írás mutató pozíciója a fájlban
- (whence: alapért. 0: abszolút poz., 1: relatív aktuális poz.-hoz, 2: rel. a fájl végéhez)

Bináris fájl és a SEEK

```
import struct

packer = struct.Struct('i3si')

with open('dates.bin', 'wb') as f:
    for i in range(5):
        values = (2020+i, b'jan', 10+i)
        packed_data = packer.pack(*values)
        f.write(packed_data)
```

```
import struct

packer = struct.Struct('i3si')

with open('dates.bin', 'rb') as f:
    f.seek(packer.size*4)
    data = f.read(packer.size)
    print(packer.unpack(data))
```

```
### output: (2024, b'jan', 14)
```

Bytesorozat vs string

- String → Byte sorozat

```
import struct
str = "hello"
str.encode()           # b'hello'

Struct.pack('8s', str)   #b'hello\x00\x00\x00'
```

- Byte sorozta → String

```
import struct
d = Struct.pack('8s', str)   #b'hello\x00\x00\x00'

d.decode().strip('\x00')    #'hello'
```

Python socket, host név feloldás

- Socket csomag használata

```
import socket
```

- `gethostname()`

```
hostname = socket.gethostname()
```

- `gethostbyname()`

```
hostip = socket.gethostbyname('www.example.org')
```

- `gethostbyname_ex()`

```
hostname, aliases, addresses = socket.gethostbyname_ex(host)
```

- `gethostbyaddr()`

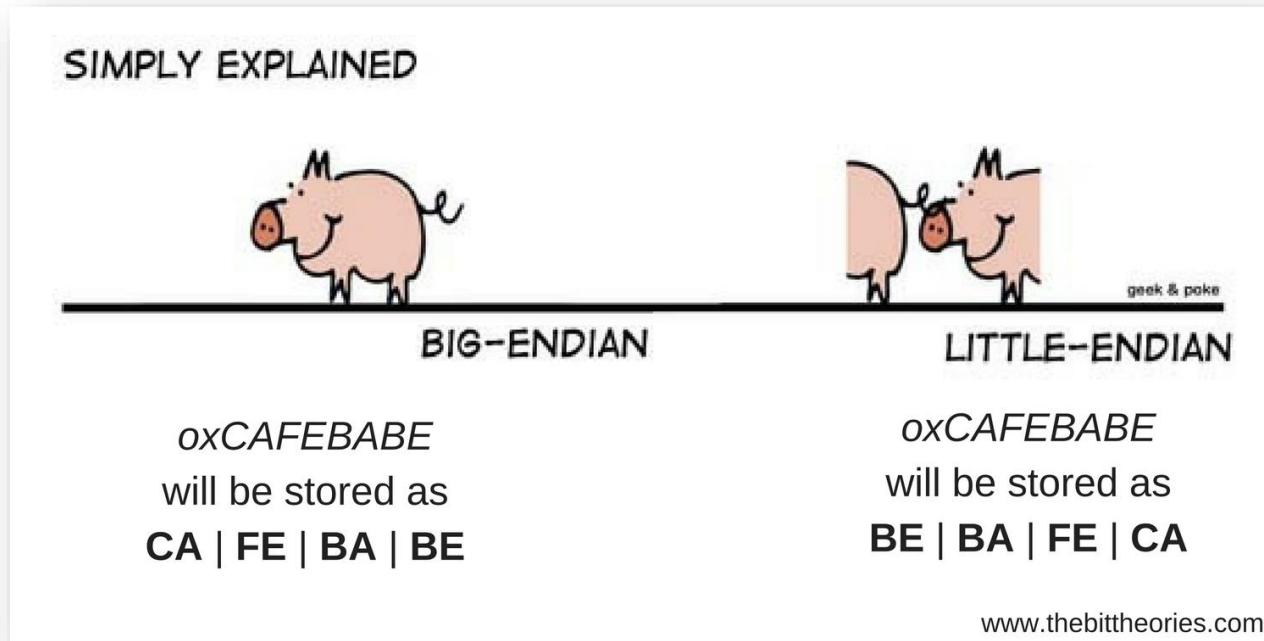
```
hostname, aliases, addrs = socket.gethostbyaddr('157.181.161.79')
```

Port számok

- Bizonyos protokollokhoz tartoznak fix portszámok, konstansok (szállítási protokollok)!
- `getservbyport()`

```
socket.getservbyport(22)
```
- Írassuk ki a 1..100-ig a portokat és a hozzájuk tartozó protokollokat!

Little endian, big endian



- 16 és 32 bites pozitív számok kódolása
 - htons(), htonl() – host to network short / long
 - ntohs(), ntohl() – network to host short / long

Tesztrendszer státusz oldal

<http://oktnb147.inf.elte.hu/ggombos/networkTester/>

Telekommunikációs hálózatok Teszt Szerver állapota

● (Last Heartbeat: 2023-09-18 08:25:52.558222, 299.44 sec ago)

Queue size: 0 estimated time to finish: 0 sec

Homeworks runtime statistics

Name	Min (sec)	Max (sec)	Avg (sec)
Barkóba	57.79	630.43	397.13
MiniNet	11	11.58	11.16
NetCopy	16.9	149.49	104.63
Protokoll	11.09	11.71	11.24
Áramkör	0	17.28	2.14

← Fut-e a tesztelő

← Mennyi beadott házi van és kb mennyi idő kell, hogy fel legyenek dolgozva?

← Statisztika az egyes házik lefutási idejéről

Feladat 1

- Adott egy bináris file, aminek a struktúrája a következő:
 - Domain (20s), port (i)
- Írjunk python scriptet, aminek parameterere:
 - port 2 # 2 sor portról megmondja milyen service tartozik hozzá
 - domain 3 # 3 sorból kiveszi a domaint és lekérdezi az ip címét
 - Ha nincs paraméter, akkor a saját domain nevünket adja meg

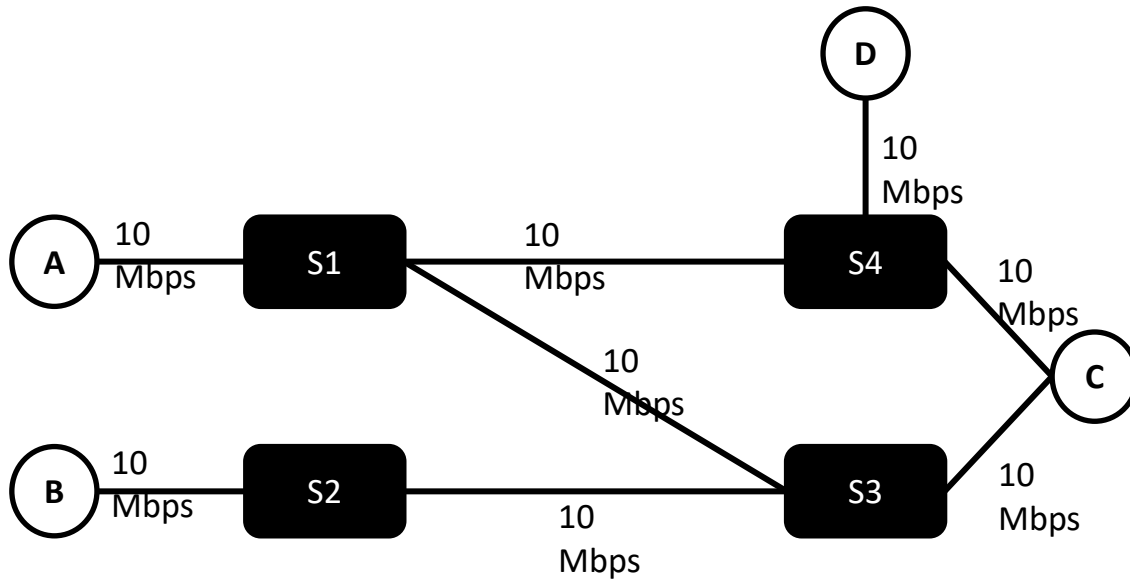
Áramkörkapcsolt hálózatok

HÁZI FELADAT I.

Feladat

<https://ggombos.web.elte.hu/halobeadando/1-aramkor/>

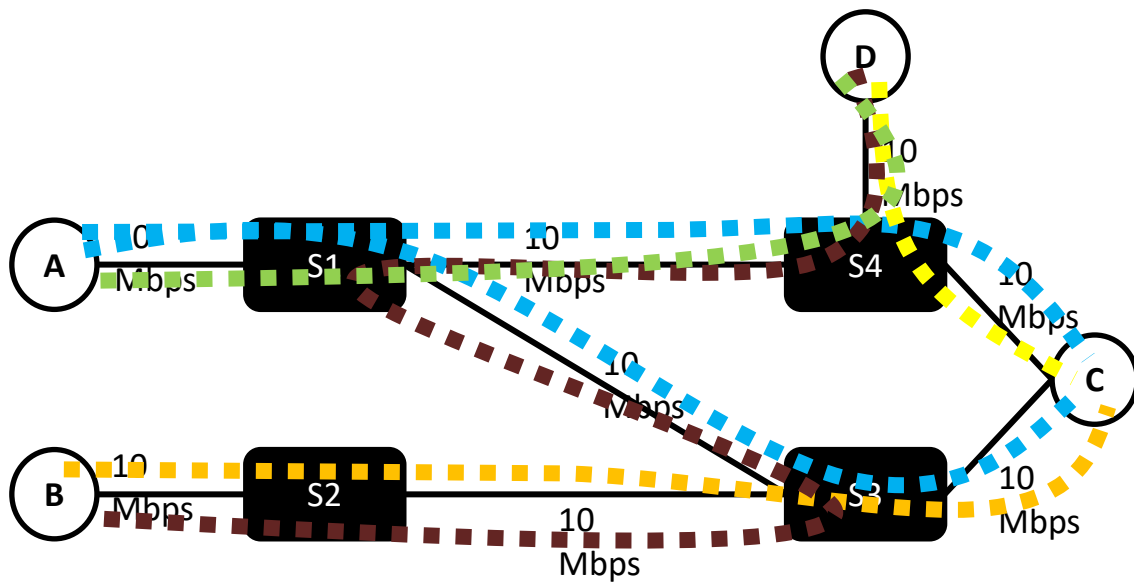
Topológia – cs1.json



Írányítatlan legyen a gráf!!!

```
"end-points": [ "A", "B", "C", "D" ],
"switches": [ "S1", "S2", "S3", "S4" ],
"links" : [
  {
    "points" : [ "A", "S1" ],
    "capacity" : 10.0
  },
  {
    "points" : [ "B", "S2" ],
    "capacity" : 10.0
  },
  {
    "points" : [ "D", "S4" ],
    "capacity" : 10.0
  },
  {
    "points" : [ "S1", "S4" ],
    "capacity" : 10.0
  },
  {
    "points" : [ "S1", "S3" ],
    "capacity" : 10.0
  },
  {
    "points" : [ "S2", "S3" ],
    "capacity" : 10.0
  },
  {
    "points" : [ "S4", "C" ],
    "capacity" : 10.0
  },
  {
    "points" : [ "S3", "C" ],
    "capacity" : 10.0
  }
]
```

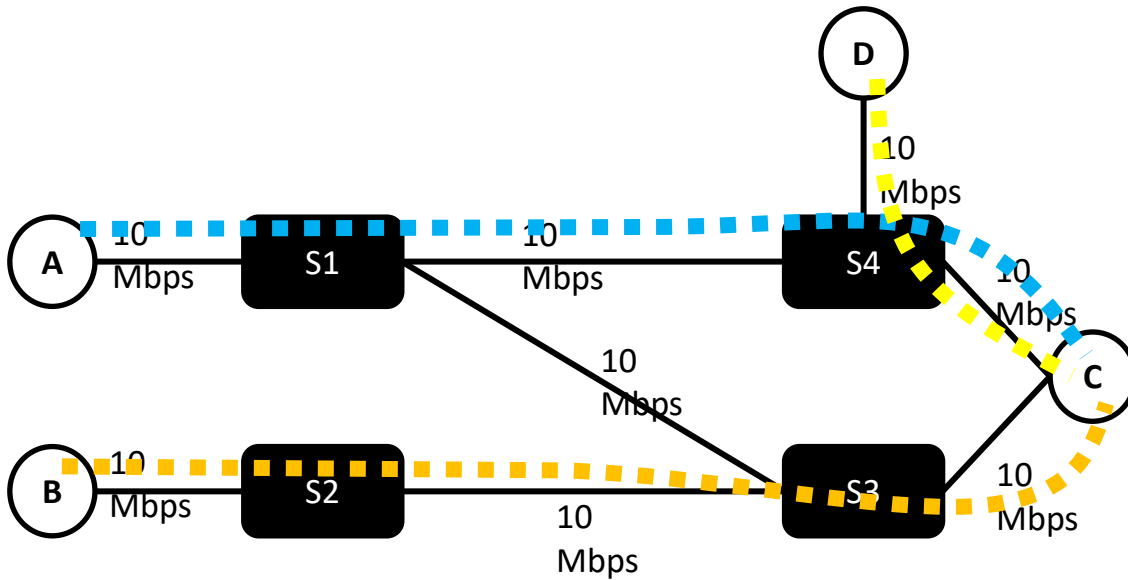
Lehetséges áramkörök – cs1.json



```
"possible-circuits" : [
  [ "D", "S4", "C" ],
  [ "C", "S4", "D" ],
  [ "A", "S1", "S4", "C" ],
  [ "C", "S4", "S1", "A" ],
  [ "B", "S2", "S3", "C" ],
  [ "C", "S3", "S2", "B" ],
  [ "A", "S1", "S3", "S4", "C" ],
  [ "C", "S4", "S3", "S1", "A" ],
  [ "B", "S2", "S3", "S4", "C" ],
  [ "C", "S4", "S3", "S2", "B" ],
  [ "A", "S1", "S4", "C" ],
  [ "C", "S4", "S1", "A" ]
]
```

Írányítatlan legyen a gráf!!!

Igények – cs1.json



Irányítatlan legyen a gráf!!!

```
"simulation" : {  
  "duration" : 11,  
  "demands" : [  
    {  
      "start-time" : 1,  
      "end-time" : 5,  
      "end-points" : ["A", "C"],  
      "demand" : 10.0  
    },  
    {  
      "start-time" : 2,  
      "end-time" : 10,  
      "end-points" : ["B", "C"],  
      "demand" : 10.0  
    },  
    {  
      "start-time" : 6,  
      "end-time" : 10,  
      "end-points" : ["D", "C"],  
      "demand" : 10.0  
    }  
  ]  
}
```

VÉGE
KÖSZÖNÖM A FIGYELMET!