PPaDS MMXXI

Matúš Jókay, C-503, matus.jokay@stuba.sk

uim.fei.stuba.sk/predmet/i-ppds

konzultácie elektronicky

mail, discord

- Problém fajčiarov
- Scoreboard
- Problém hodujúcich divochov

Problém fajčiarov

Problém fajčiarov

- 4 vlákna

Problém fajčiarov

- 4 vlákna
  - Agent
  - Traja fajčiari

Problém fajčiarov

- 4 vlákna
  - Agent
  - Traja fajčiari
- Fajčiari v nekonečnej slučke

Problém fajčiarov

- 4 vlákna
  - Agent
  - Traja fajčiari
- Fajčiari v nekonečnej slučke
  - Najprv čakajú na materiál pre ušúľanie cigaretle
  - Keď má fajčiar materiál, ušúľa si cigaretu a užije ju

Problém fajčiarov

- Agent má nekonečné množstvo 3 ingrediencií

Problém fajčiarov

- Agent má nekonečné množstvo 3 ingrediencií
  - Tabak
  - Papier
  - Zápalky

Problém fajčiarov

- Agent má nekonečné množstvo 3 ingrediencií
  - Tabak
  - Papier
  - Zápalky
- Každý z fajčiarov má k dispozícii nekonečné množstvo iba jednej z 3 surovín!

Problém fajčiarov

- Agent má nekonečné množstvo 3 ingrediencií
  - Tabak
  - Papier
  - Zápalky
- Každý z fajčiarov má k dispozícii nekonečné množstvo iba jednej z 3 surovín!
- Jeden má tabak, druhý papier, tretí zápalky

Problém fajčiarov

- Agent v nekonečnom cykle

Problém fajčiarov

- Agent v nekonečnom cykle
  - Náhodne vyberie 2 z 3 surovín
  - Dá ich k dispozícii fajčiarom

Problém fajčiarov

- Agent v nekonečnom cykle
  - Náhodne vyberie 2 z 3 surovín
  - Dá ich k dispozícii fajčiarom
- Podľa toho, aké suroviny agent dal k dispozícii

Problém fajčiarov

- Agent v nekonečnom cykle
  - Náhodne vyberie 2 z 3 surovín
  - Dá ich k dispozícii fajčiarom
- Podľa toho, aké suroviny agent dal k dispozícii
  - Mal by sa chopiť aktivity ten pravý fajčiar
  - Zobrať suroviny a pokračovat (ušúľať cigaretu a fajčiť)

Problém fajčiarov

- Príklad

Problém fajčiarov

- Príklad
- Agent dal k dispozícii tabak a papier

Problém fajčiarov

- Príklad
- Agent dal k dispozícii tabak a papier
- Preto fajčiar, ktorý má nekonečný zdroj zápaliek

Problém fajčiarov

- Príklad
- Agent dal k dispozícii tabak a papier
- Preto fajčiar, ktorý má nekonečný zdroj zápaliek
  - Zoberie tabak a papier

Problém fajčiarov

- Príklad
- Agent dal k dispozícii tabak a papier
- Preto fajčiar, ktorý má nekonečný zdroj zápaliek
  - Zoberie tabak a papier
  - Ušúľa cigaretu (a fajčí)

Problém fajčiarov

- Príklad
- Agent dal k dispozícii tabak a papier
- Preto fajčiar, ktorý má nekonečný zdroj zápaliek
  - Zoberie tabak a papier
  - Ušúľa cigaretu (a fajčí)
  - Dá signál agentovi, že suroviny spracoval

Problém fajčiarov

- Aká to má vzťah k realite virtuálneho sveta počítačov?

Problém fajčiarov

- Aká to má vzťah k realite virtuálneho sveta počítačov?

- Agent je OS, ktorý prerozdeľuje zdroje
- Fajčiari sú aplikácie, ktoré zdroje potrebujú

Problém fajčiarov

- Agent – OS, fajčiari – aplikácie
- Synchronizačný problém

Problém fajčiarov

- Agent – OS, fajčiari – aplikácie
- Synchronizačný problém
  - Ak nejaké voľné zdroje umožňujú, aby nejaká aplikácia pokračovala vo svojej činnosti, táto aplikácia by sa mala "zobudiť" a pokračovať vo svojej práci

Problém fajčiarov

- Agent – OS, fajčiari – aplikácie
- Synchronizačný problém
  - Ak nejaké voľné zdroje umožňujú, aby nejaká aplikácia pokračovala vo svojej činnosti, táto aplikácia by sa mala "zobudiť" a pokračovať vo svojej práci
  - Opačné garde: nesmie sa zobudiť žiadna aplikácia, ktorá (kvôli nedostatku zdrojov, ktoré potrebuje) nemôže pokračovať vo svojej práci

Problém fajčiarov

- 3 najčastejšie verzie tohto problému v literatúre

Problém fajčiarov

- 3 najčastejšie verzie tohto problému v literatúre

1. Nemožná verzia
2. Zaujímavá verzia
3. Triviálna verzia

Problém fajčiarov

- Nemožná verzia

Problém fajčiarov

- Nemožná verzia
- Nemožno meniť kód agenta

Problém fajčiarov

- Nemožná verzia
- Nemožno meniť kód agenta (asi bežne nemeníme kód jadra OS, či?)

Problém fajčiarov

- Nemožná verzia

- Nemožno meniť kód agenta
- Nemožno používať if-else alebo pole semaforov

Problém fajčiarov

- Nemožná verzia

- Nemožno meniť kód agenta
- Nemožno používať if-else alebo pole semaforov

- Názov vystihuje riešenie...

Problém fajčiarov

- Nemožná verzia

- Nemožno meniť kód agenta
- Nemožno používať if-else alebo pole semaforov

- Názov vystihuje riešenie... Žiadne nejestvuje

Problém fajčiarov

- Zaujímavá verzia

Problém fajčiarov

- Zaujímavá verzia
- Nemožno meniť kód agenta (nechceme meniť OS)

Problém fajčiarov

- Zaujímavá verzia

- Nemožno meniť kód agenta (nechceme meniť OS)
- Môžeme používať if-else aj semafory

Problém fajčiarov

- Triviálna verzia

Problém fajčiarov

- Triviálna verzia

- Agent podľa toho, čo vybral, signalizuje správnemu fajčiarovi, aby pokračoval

Problém fajčiarov

- Triviálna verzia

- Agent podľa toho, čo vybral, signalizuje správnemu fajčiarovi, aby pokračoval
  - Agent musí vedieť, na čo to-ktoré vlákno čaká

Problém fajčiarov

- Triviálna verzia

- Agent podľa toho, čo vybral, signalizuje správnemu fajčiarovi, aby pokračoval
  - Agent musí vedieť, na čo to-ktoré vlákno čaká
  - Jadro problému (ingrediencie a cigarety) postráda zmysel
  - Problém sa redukuje na triviálnu synchronizáciu

Problém fajčiarov

- Agenta budeme modelovať pomocou 3 vlákien

Problém fajčiarov

- Agenta budeme modelovať pomocou 3 vlákien
- Každé vlákno bude poskytovať iné dve suroviny

Problém fajčiarov

- Agenta budeme modelovať pomocou 3 vlákien
- Každé vlákno bude poskytovať iné dve suroviny
- Jedná sa o 3 konkurentné vlákna

Problém fajčiarov

- Agenta budeme modelovať pomocou 3 vlákien
- Každé vlákno bude poskytovať iné dve suroviny
- Jedná sa o 3 konkurentné vlákna, ale vždy iba 1 môže byť aktívne!

Problém fajčiarov

- Agenta budeme modelovať pomocou 3 vlákien
- Každé vlákno bude poskytovať iné dve suroviny
- Jedná sa o 3 konkurentné vlákna, ale vždy iba 1 môže byť aktívne! → použijeme Multiplex(1)

Problém fajčiarov

- Agenta budeme modelovať pomocou 3 vlákien
- Každé vlákno bude poskytovať iné dve suroviny
- Jedná sa o 3 konkurentné vlákna, ale vždy iba 1 môže byť aktívne! → použijeme Multiplex(1)

- !!! Pozor na modelovanie s FIFO semaforom !!!

Problém fajčiarov

```
Agent_A():
1)  agentSem.wait()
2)  tobacco.signal()
3)  paper.signal()
```

```
Agent_B():
1)  agentSem.wait()
2)  paper.signal()
3)  match.signal()
```

```
Agent_C():
1)  agentSem.wait()
2)  tobacco.signal()
3)  match.signal()
```

```
Init_agent():
1)  agentSem = Semaphore(1)

2)  tobacco = Semaphore(0)
3)  paper = Semaphore(0)
4)  match = Semaphore(0)
```

Problém fajčiarov #1

```
Smoker_M():
1)  tobacco.wait()
2)  paper.wait()
3)  agentSem.signal()
```

```
Smoker_T():
1)  paper.wait()
2)  match.wait()
3)  agentSem.signal()
```

```
Smoker_P():
1)  tobacco.wait()
2)  match.wait()
3)  agentSem.signal()
```

Problém fajčiarov #1

Agent_A():
1) agentSem.wait()
2) tobacco.signal()
3) paper.signal()

Agent_B():
1) agentSem.wait()
2) paper.signal()
3) match.signal()

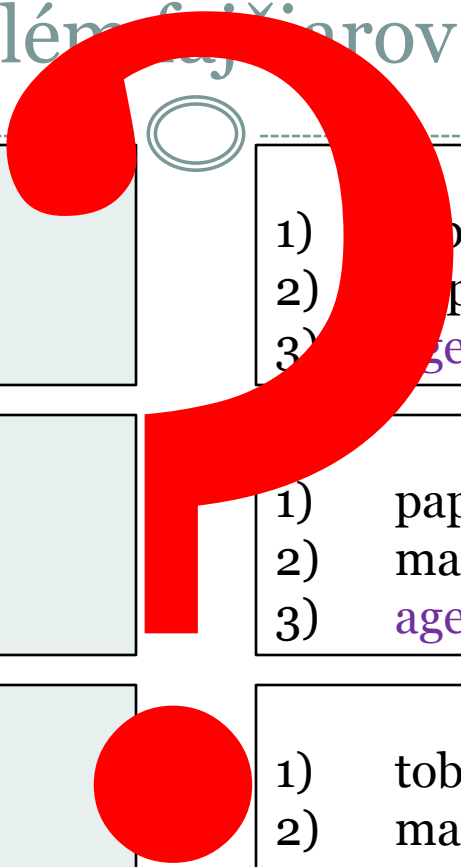
Agent_C():
1) agentSem.wait()
2) tobacco.signal()
3) match.signal()

Smoker_M():
1) tobacco.wait()
2) paper.wait()
3) agentSem.signal()

Smoker_T():
1) paper.wait()
2) match.wait()
3) agentSem.signal()

Smoker_P():
1) tobacco.wait()
2) match.wait()
3) agentSem.signal()

Problém fajčiarov #1



Agent_A():
1) agentSem.wait()
2) tobacco.signal()
3) paper.signal()

Agent_B():
1) agentSem.wait()
2) paper.signal()
3) match.signal()

Agent_C():
1) agentSem.wait()
2) tobacco.signal()
3) match.signal()

Smoker_M():
1) bacco.wait()
2) per.wait()
3) gentSem.signal()

Smoker_T():
1) paper.wait()
2) match.wait()
3) agentSem.signal()

Smoker_P():
1) tobacco.wait()
2) match.wait()
3) agentSem.signal()

Problém fajčiarov #1

Agent_A():
1) agentSem.wait()
2) tobacco.signal()
3) paper.signal()

Agent_B():
1) agentSem.wait()
2) paper.signal()
3) match.signal()

Agent_C():
1) agentSem.wait()
2) tobacco.signal()
3) match.signal()

Smoker_M():
1) tobacco.wait()
2) paper.wait()
3) agentSem.signal()

Smoker_T():
1) paper.wait()
2) match.wait()
3) agentSem.signal()

Smoker_P():
1) tobacco.wait()
2) match.wait()
3) agentSem.signal()

Problém fajčiarov #1

```
Agent_A():
1) agentSem.wait()
2) tobacco.signal()
3) paper.signal()
```

```
Smoker_M():
1) tobacco.wait()
2) paper.wait()
3) agentSem.signal()
```

```
Agent_B():
1) agentSem.wait()
2) paper.signal()
3) match.signal()
```

```
Smoker_T():
1) paper.wait()
2) match.wait()
3) agentSem.signal()
```

```
Agent_C():
1) agentSem.wait()
2) tobacco.signal()
3) match.signal()
```

```
Smoker_P():
1) tobacco.wait()
2) match.wait()
3) agentSem.signal()
```

Problém fajčiarov #1

```
Agent_A():
1)  agentSem.wait()
2)  tobacco.signal()
3)  paper.signal()
```

```
Smoker_M():
1)  tobacco.wait()
2)  paper.wait()
3)  agentSem.signal()
```

```
Agent_B():
1)  agentSem.wait()
2)  paper.signal()
3)  match.signal()
```

```
Smoker_T():
1)  paper.wait()
2)  match.wait()
3)  agentSem.signal()
```

```
Agent_C():
1)  agentSem.wait()
2)  tobacco.signal()
3)  match.signal()
```

```
Smoker_P():
1)  tobacco.wait()
2)  match.wait()
3)  agentSem.signal()
```

Problém fajčiarov #1

```
Agent_A():
1) agentSem.wait()
2) tobacco.signal()
3) paper.signal()
```

```
Smoker_M():
1) tobacco.wait()
2) paper.wait()
3) agentSem.signal()
```

```
Agent_B():
1) agentSem.wait()
2) paper.signal()
3) match.signal()
```

```
Smoker_T():
1) paper.wait()
2) match.wait()
3) agentSem.signal()
```

```
Agent_C():
1) agentSem.wait()
2) tobacco.signal()
3) match.signal()
```

```
Smoker_P():
1) tobacco.wait()
2) match.wait()
3) agentSem.signal()
```

Problém fajčiarov #1

```
Agent_A():
1) agentSem.wait()
2) tobacco.signal()
3) paper.signal()
```

```
Agent_B():
1) agentSem.wait()
2) paper.signal()
3) match.signal()
```

```
Agent_C():
1) agentSem.wait()
2) tobacco.signal()
3) match.signal()
```

```
Smoker_M():
1) tobacco.wait()
2) paper.wait()
3) agentSem.signal()
```

```
Smoker_T():
1) paper.wait()
2) match.wait()
3) agentSem.signal()
```

```
Smoker_P():
1) tobacco.wait()
2) match.wait()
3) agentSem.signal()
```

Problém fajčiarov #1

```
Agent_A():
1) agentSem.wait()
2) tobacco.signal()
3) paper.signal()
```

```
Agent_B():
1) agentSem.wait()
2) paper.signal()
3) match.signal()
```

```
Agent_C():
1) agentSem.wait()
2) tobacco.signal()
3) match.signal()
```

```
Smoker_M():
1) tobacco.wait()
2) paper.wait()
3) agentSem.signal()
```

```
Smoker_T():
1) paper.wait()
2) match.wait()
3) agentSem.signal()
```

```
Smoker_P():
1) tobacco.wait()
2) match.wait()
3) agentSem.signal()
```

Problém fajčiarov #1

```
Agent_A():
1) agentSem.wait()
2) tobacco.signal()
3) paper.signal()
```

```
Agent_B():
1) agentSem.wait()
2) paper.signal()
3) match.signal()
```

```
Agent_C():
1) agentSem.wait()
2) tobacco.signal()
3) match.signal()
```

```
Smoker_M():
1) tobacco.wait()
2) paper.wait()
3) agentSem.signal()
```

```
Smoker_T():
1) paper.wait()
2) match.wait()
3) agentSem.signal()
```

```
Smoker_P():
1) tobacco.wait()
2) match.wait()
3) agentSem.signal()
```

Problém fajčiarov #1

```
Agent_A():
1) agentSem.wait()
2) tobacco.signal()
3) paper.signal()
```

```
Agent_B():
1) agentSem.wait()
2) paper.signal()
3) match.signal()
```

```
Agent_C():
1) agentSem.wait()
2) tobacco.signal()
3) match.signal()
```

```
Smoker_M():
1) tobacco.wait()
2) paper.wait()
3) agentSem.signal()
```

```
Smoker_T():
1) paper.wait()
2) match.wait()
3) agentSem.signal()
```

```
Smoker_P():
1) tobacco.wait()
2) match.wait()
3) agentSem.signal()
```

Problém fajčiarov

- Naivné riešenie nie je správne
- Uviaznutie

Problém fajčiarov

- Naivné riešenie nie je správne
- Uviaznutie

- **Problém je zaujímavý tým, že ho nevieme riešiť priamo len pomocou agenta a fajčiarov**

Problém fajčiarov

- Naivné riešenie nie je správne
- Uviaznutie

- Problém je zaujímavý tým, že ho nevieme riešiť priamo len pomocou agenta a fajčiarov
- Potrebujeme **pomocných agentov** pre fajčiarov

Problém fajčiarov #2

- Pomocný agent fajčiara – díler

Problém fajčiarov #2

- Pomocný agent fajčiara – díler
- Stará sa o “dodávku” materiálu pre konkrétneho fajčiara

Problém fajčiarov #2

- Pomocný agent fajčiara – díler
- Stará sa o “dodávku” materiálu pre konkrétneho fajčiara
- Má na starosti prebudenie fajčiara, o ktorého sa stará

Problém fajčiarov #2

- Pomocný agent fajčiara – díler
- Stará sa o “dodávku” materiálu pre konkrétneho fajčiara
- Má na starosti prebudenie fajčiara, o ktorého sa stará
- Sleduje stav “trhu”, a podľa toho buď zobudí alebo nezobudí svojho fajčiara

Problém fajčiarov #2

- Pomocný agent fajčiara – díler v programe

Problém fajčiarov #2

- Pomocný agent fajčiara – díler v programe
- Reaguje na signál od agenta

Problém fajčiarov #2

- Pomocný agent fajčiara – díler v programe

- Reaguje na signál od agenta
- Sleduje stav jednotlivých tovarov

Problém fajčiarov #2

- Pomocný agent fajčiara – díler v programe

- Reaguje na signál od agenta
- Sleduje stav jednotlivých tovarov
- Budí fajčiara, ktorý môže pokračovat

Problém fajčiarov #2

```
Init_agent():
1)    agentSem = Semaphore(1)

2)    tobacco = Semaphore(0)
3)    paper = Semaphore(0)
4)    match = Semaphore(0)
```

Problém fajčiarov #2

Smoker_pushers_init():

1) isTobacco = False
2) isMatch = False
3) isPaper = False

4) tobaccoSmo = Semaphore(0)
5) paperSmo = Semaphore(0)
6) matchSmo = Semaphore(0)

Init_agent():

1) agentSem = Semaphore(1)

2) tobacco = Semaphore(0)
3) paper = Semaphore(0)
4) match = Semaphore(0)

Problém fajčiarov #2

Smoker_pushers_init():

1) isTobacco = False
2) isMatch = False
3) isPaper = False

4) tobaccoSmo = Semaphore(0)
5) paperSmo = Semaphore(0)
6) matchSmo = Semaphore(0)

- premenné typu bool – či je daný tovar na stole (dostupný)
- tobaccoSmo – signalizuje fajčiarovi s tabakom, že môže pokračovať; podobne ostatné semafory

Problém fajčiarov #2

Smoker_T():

1) tobaccoSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

- premenné typu bool – či je daný tovar na stole (dostupný)
- tobaccoSmo – signalizuje fajčiarovi s tabakom, že môže pokračovať; podobne ostatné semafory

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

1.
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

1. tobacco.wait()
2.
3.
4.
5.
6.
7.
8.
9.
10.
11.

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

```
1.      tobacco.wait()
2.
3.          if isPaper:
4.
5.
6.
7.
8.
9.
10.
11.
```

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

```
1.      tobacco.wait()
2.
3.          if isPaper:
4.              isPaper ← False
5.
6.
7.
8.
9.
10.
11.
```

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

```
1.      tobacco.wait()
2.
3.          if isPaper:
4.              isPaper ← False
5.              matchSmo.signal()
6.
7.
8.
9.
10.
11.
```

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

```
1.      tobacco.wait()
2.
3.          if isPaper:
4.              isPaper ← False
5.              matchSmo.signal()
6.          elif isMatch:
7.
8.
9.
10.
11.
```

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

```
1.      tobacco.wait()
2.
3.          if isPaper:
4.              isPaper ← False
5.              matchSmo.signal()
6.          elif isMatch:
7.              isMatch ← False
8.
9.
10.
11.
```

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

```
1.      tobacco.wait()
2.
3.          if isPaper:
4.              isPaper ← False
5.              matchSmo.signal()
6.          elif isMatch:
7.              isMatch ← False
8.              paperSmo.signal()
9.
10.
11.
```

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

```
1.    tobacco.wait()
2.
3.      if isPaper:
4.          isPaper ← False
5.          matchSmo.signal()
6.      elif isMatch:
7.          isMatch ← False
8.          paperSmo.signal()
9.      else:
10.         isTobacco ← True
11.
```

Problém fajčiarov #2

Smoker_M():

1) matchSmo.wait()
2) makeCigarette()
3) agentSem.signal()
4) smoke()

Pusher_A():

```
1.  tobacco.wait()
2.  mutex.wait()
3.      if isPaper:
4.          isPaper ← False
5.          matchSmo.signal()
6.      elif isMatch:
7.          isMatch ← False
8.          paperSmo.signal()
9.      else:
10.         isTobacco ← True
11. mutex.signal()
```

Problém fajčiarov

- Agent musí stále čakať
- Nemôže pokračovať, pokým nezmizne dodaný tovar „zo stola“

```
Agent_XYZ():
1)    agentSem.wait()
2)    ingredient1.signal()
3)    ingredient2.signal()
```

- Vieme vylepšiť riešenie tak, aby agent nečakal?

Problém fajčiarov #3

Smoker_pushers_init():

1) isTobacco = False
2) isMatch = False
3) isPaper = False

4) tobaccoSmo = Semaphore(0)
5) paperSmo = Semaphore(0)
6) matchSmo = Semaphore(0)

Problém fajčiarov #3

Smoker_pushers_init():

1) isTobacco = 0
2) isMatch = 0
3) isPaper = 0

4) tobaccoSmo = Semaphore(0)
5) paperSmo = Semaphore(0)
6) matchSmo = Semaphore(0)

Problém fajčiarov #3

Smoker_pushers_init():

1) isTobacco = 0
2) isMatch = 0
3) isPaper = 0

4) tobaccoSmo = Semaphore(0)
5) paperSmo = Semaphore(0)
6) matchSmo = Semaphore(0)

Problém fajčiarov #3

Smoker_pushers_init():

1) isTobacco = 0
2) isMatch = 0
3) isPaper = 0

4) tobaccoSmo = Semaphore(0)
5) paperSmo = Semaphore(0)
6) matchSmo = Semaphore(0)

Pusher_A():

```
1.  tobacco.wait()
2.  mutex.wait()
3.      if isPaper:
4.          isPaper ← False
5.          matchSmo.signal()
6.      elif isMatch:
7.          isMatch ← False
8.          paperSmo.signal()
9.      else:
10.         isTobacco ← True
11. mutex.signal()
```

Problém fajčiarov #3

Smoker_pushers_init():

1) isTobacco = 0
2) isMatch = 0
3) isPaper = 0

4) tobaccoSmo = Semaphore(0)
5) paperSmo = Semaphore(0)
6) matchSmo = Semaphore(0)

Pusher_A():

```
1.  tobacco.wait()
2.  mutex.wait()
3.      if isPaper:
4.          isPaper ← False
5.          matchSmo.signal()
6.      elif isMatch:
7.          isMatch ← False
8.          paperSmo.signal()
9.      else:
10.         isTobacco ← True
11. mutex.signal()
```

Problém fajčiarov #3

Smoker_pushers_init():

1) isTobacco = 0
2) isMatch = 0
3) isPaper = 0

4) tobaccoSmo = Semaphore(0)
5) paperSmo = Semaphore(0)
6) matchSmo = Semaphore(0)

Pusher_A():

```
1.  tobacco.wait()
2.  mutex.wait()
3.      if isPaper:
4.          isPaper -= 1
5.          matchSmo.signal()
6.      elif isMatch:
7.          isMatch -= 1
8.          paperSmo.signal()
9.      else:
10.         isTobacco += 1
11. mutex.signal()
```

Problém fajčiarov #3

Smoker_pushers_init():

1) isTobacco = 0
2) isMatch = 0
3) isPaper = 0

4) tobaccoSmo = Semaphore(0)
5) paperSmo = Semaphore(0)
6) matchSmo = Semaphore(0)

Pusher_A():

```
1.  tobacco.wait()
2.  mutex.wait()
3.      if isPaper:
4.          isPaper -= 1
5.          matchSmo.signal()
6.      elif isMatch:
7.          isMatch -= 1
8.          paperSmo.signal()
9.      else:
10.         isTobacco += 1
11. mutex.signal()
```

Problém fajčiarov - vizualizácia

- Majme miestnosť

Problém fajčiarov - vizualizácia

- Majme miestnosť:
  - Stôl
  - Na stoličkách spia fajčiari ;)

Problém fajčiarov - vizualizácia

- Majme miestnosť:
  - Stôl
  - Na stoličkách spia fajčiari ;)
- Agent poveruje dílerov dodávaním surovín

Problém fajčiarov - vizualizácia

- Majme miestnosť:
  - Stôl
  - Na stoličkách spia fajčiari ;)
- Agent poveruje dílerov dodávaním surovín
- Díler

Problém fajčiarov - vizualizácia

- Majme miestnosť:
  - Stôl
  - Na stoličkách spia fajčiari ;)
- Agent poveruje dílerov dodávaním surovín
- Díler (1 díler nosí do miestnosti iba 1 surovinu!)

Problém fajčiarov - vizualizácia

- Majme miestnosť:
  - Stôl
  - Na stoličkách spia fajčiari ;)
- Agent poveruje dílerov dodávaním surovín
- Díler (1 díler nosí do miestnosti iba 1 surovinu!)
  - Po JEDNOM chodia do miestnosti

Problém fajčiarov - vizualizácia

- Majme miestnosť:
  - Stôl
  - Na stoličkách spia fajčiari ;)
- Agent poveruje dílerov dodávaním surovín
- Díler (1 díler nosí do miestnosti iba 1 surovinu!)
  - Po JEDNOM chodia do miestnosti
  - Skontroluje stôl, či môže skompletizovať dodávku pre fajč.

Problém fajčiarov - vizualizácia

- Majme miestnosť:
  - Stôl
  - Na stoličkách spia fajčiari ;)
- Agent poveruje dílerov dodávaním surovín
- Díler (1 díler nosí do miestnosti iba 1 surovinu!)
  - Po JEDNOM chodia do miestnosti
  - Skontroluje stôl, či môže skompletizovať dodávku pre fajč.
  - Ak áno, zobudí ho; ak nie, nechá surovinu na stole a odíde

Scoreboard

Scoreboard

- Majme miestnosť

Scoreboard

- Majme miestnosť:
  - Stôl so surovinami

Scoreboard

- Majme miestnosť:
  - Stôl so surovinami
  - Spiacich ľudí

Scoreboard

- Majme miestnosť:
  - Stôl so surovinami
  - Spiacich ľudí
- Díler

Scoreboard

- Majme miestnosť:
  - Stôl so surovinami
  - Spiacich ľudí
- Díler
  - Vojde do miestnosti (díleri vchádzajú po jednom!!!)

Scoreboard

- Majme miestnosť:
  - Stôl so surovinami
  - Spiacich ľudí
- Díler
  - Vojde do miestnosti (díleri vchádzajú po jednom!!!)
  - Skontroluje stav surovín na stole

Scoreboard

- Majme miestnosť:
  - Stôl so surovinami
  - Spiacich ľudí
- Díler
  - Vojde do miestnosti (díleri vchádzajú po jednom!!!)
  - Skontroluje stav surovín na stole
  - Na základe stavu vyvolá aktivitu

Scoreboard

- Majme miestnosť:
  - Stôl so surovinami
  - Spiacich ľudí
- Díler
  - Vojde do miestnosti (díleri vchádzajú po jednom!!!)
  - Skontroluje stav surovín na stole
  - Na základe stavu vyvolá aktivitu (napr. zobudí nejakého spáča)

Scoreboard

- Scoreboard – ktorý budeme možno ešte aj inde využívať

Scoreboard

- Scoreboard – ktorý budeme možno ešte aj inde využívať

- Číselné premenné reprezentujú stav systému

Scoreboard

- Scoreboard – ktorý budeme možno ešte aj inde využívať

- Číselné premenné reprezentujú stav systému
- Vlákna po jednom (cez mutex) kontrolujú stav systému

Scoreboard

- Scoreboard – vzor, ktorý budeme možno ešte aj inde využívať

- Číselné premenné reprezentujú stav systému
- Vlákna po jednom (cez mutex) kontrolujú stav systému
- Na základe stavu vyvolávajú želanú aktivitu

Problém hodujúcich divochov

Problém hodujúcich divochov

- Kmeň divochov má klasické hody

Problém hodujúcich divochov

- Kmeň divochov má klasické hody
- Jeden spoločný hrniec, z ktorého si každý sám berie (ak hrniec nie je prázdny!)

Problém hodujúcich divochov

- Kmeň divochov má klasické hody
- Jeden spoločný hrniec, z ktorého si každý sám berie (ak hrniec nie je prázdny!)
- Hrniec dokáže poňať M porcií duseného misionára

Problém hodujúcich divochov

- Kmeň divochov má klasické hody
- Jeden spoločný hrniec, z ktorého si každý sám berie (ak hrniec nie je prázdny!)
- Hrniec dokáže poňať M porcií duseného misionára
- Ak je hrniec prázdny

Problém hodujúcich divochov

- Kmeň divochov má klasické hody
- Jeden spoločný hrniec, z ktorého si každý sám berie (ak hrniec nie je prázdny!)
- Hrniec dokáže poňať M porcií duseného misionára
- Ak je hrniec prázdny:
  - Divoch, ktorý si práve chcel nabrať, zobudí kuchára,
  - A čaká, kým kuchár nenaplní hrniec

Problém hodujúcich divochov

Nesynchronizovaný kód
savage():

```
1)    while True:
2)        getServingFromPot()
3)        eat()
```

Nesynchronizovaný kód
cook():

```
1)    while True:
2)        putServingsInPot()
```

- Synchronizačné obmedzenia
  - Divoch nesmie zobrať porciu, ak je hrniec prázdny
  - Kuchár smie naplnit hrniec, iba ak je tento prázdny

Problém hodujúcich divochov

- Môžeme použiť prístup ako v prípade synchronizačného problému konzument-producent

Problém hodujúcich divochov

- Môžeme použiť prístup ako v prípade synchronizačného problému konzument-producent
- Použime počítalo porcií
  - Počet porcií
  - Zobudenie kuchára
  - Čakanie na signál doplnenia hrnca

init():

1) servings ← 0
2) mutex ← Mutex()
3) emptyPot ← Event()
4) fullPot ← Event()

emptyPot – hrniec je prázdny
fullPot – hrniec je plný

Problém hodujúcich divochov

savage():

1)
2)
3)
4)
5)
6)
7)
8)
9)
10)

cook():

1)
2)
3)
4)
5)
6)
7)
8)
9)
10)

Problém hodujúcich divochov

savage():

1) while True:
2)
3)
4)
5)
6)
7)
8)
9)
10)

cook():

1)
2)
3)
4)
5)
6)
7)
8)
9)
10)

Problém hodujúcich divochov

savage():

```
1)      while True:
2)
3)
4)
5)
6)
7)
8)
9)
10)         eat()
```

cook():

```
1)
2)
3)
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)
3)
4)
5)
6)
7)          getServingFromPot()
8)
9)
10)       eat()
```

cook():

```
1)
2)
3)
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)
4)
5)
6)
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)
2)
3)
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)
4)
5)
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)
2)
3)
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)
5)
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)
2)
3)
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)
2)
3)
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)
2)
3)
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

1)
2)
3)
4)
5)
6)
7)
8)
9)
10)

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)      while True:
2)
3)
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)      while True:
2)
3)              putServingsInPot()
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)              servings ← M
5)
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)              servings ← M
5)              fullPot.signal()
6)
7)
8)
9)
10)
```

Problém hodujúcich divochov

savage():

```
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)
10)         eat()
```

cook():

```
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)              servings ← M
5)              fullPot.signal()
```

Problém hodujúcich divochov

```
savage():
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)          eat()
```

```
cook():
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)              servings ← M
5)              fullPot.signal()
```

- Rôzne otázky...

Problém hodujúcich divochov

```
savage():
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)          eat()
```

```
cook():
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)              servings ← M
5)              fullPot.signal()
```

- Rôzne otázky...
  1. R4 cook()

Problém hodujúcich divochov

```
savage():
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)          eat()
```

```
cook():
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)              servings ← M
5)              fullPot.signal()
```

- Rôzne otázky...
  1. R4 cook()
  2. Môže nastať uviaznutie? V akom prípade?

Problém hodujúcich divochov

```
savage():
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)          eat()
```

```
cook():
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)              servings ← M
5)              fullPot.signal()
```

- Rôzne otázky...
  1. R4 cook()
  2. Môže nastať uviaznutie? V akom prípade?
  3. Aký je prístup do hrnca vzhľadom na vlákna?

Problém hodujúcich divochov

```
savage():
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)          eat()
```

```
cook():
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)              servings ← M
5)              fullPot.signal()
```

- Rôzne otázky…
  1. R4 cook()
  2. Môže nastať uviaznutie? V akom prípade?
  3. Aký je prístup do hrnca vzhľadom na vlákna? Sú funkcie getServing… a putServings… bezpečné?

Problém hodujúcich divochov

```
savage():
1)      while True:
2)          mutex.lock()
3)              if servings == 0:
4)                  emptyPot.signal()
5)                  fullPot.wait()
6)              servings -= 1
7)              getServingFromPot()
8)          mutex.unlock()
9)          eat()
```

```
cook():
1)      while True:
2)              emptyPot.wait()
3)              putServingsInPot()
4)              servings ← M
5)              fullPot.signal()
```

- Rôzne otázky...
  1. R4 cook()
  2. Môže nastať uviaznutie? V akom prípade?
  3. Aký je prístup do hrnca vzhľadom na vlákna? Sú funkcie getServing... a putServings... bezpečné?
  4. Musí byť getServingFromPot v KO?

Ďakujem za pozornosť