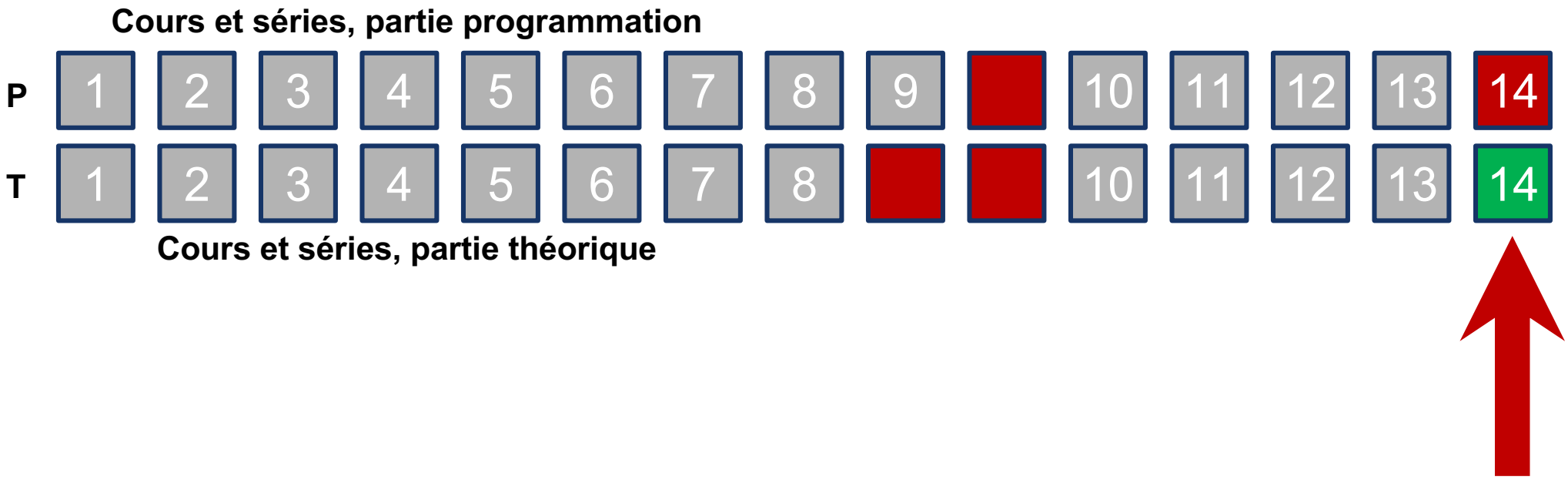
Information, Calcul et Communication

CS-119(k) ICC – Programmation
Semaine 14

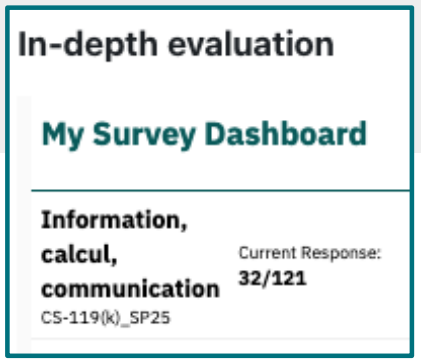
Rafael Pires
rafael.pires@epfl.ch

Lausanne, 26.05.2025

Planning


Cours et séries, partie programmation
P
1
2
3
4
5
6
7
8
9
10
11
12
13
14
T
1
2
3
4
5
6
7
8
10
11
12
13
14
Cours et séries, partie théorique
30.05 Quizz (108/120)

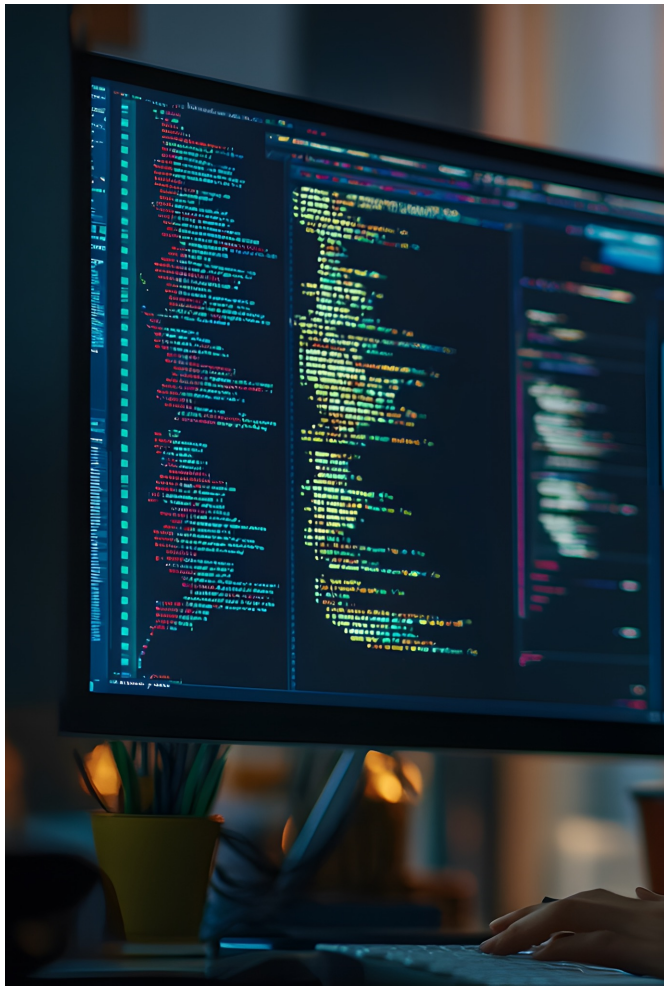
Évaluation approfondie



- Dès aujourd'hui jusqu'au dimanche 8 juin
  - Connectez-vous à Moodle et restez sur la page d'accueil (tableau de bord, pas la page du cours)
  - Cliquer sur la flèche en haut à droite de l'écran qui fera apparaître un bloc contenant la tuile intitulée « Évaluation approfondie »

- Donnez un retour sur le cours
  - Feedback anonyme
  - Dites ce qui vous a plu, ce qui vous a déplu, et vos suggestions d'amélioration
  - Soyez constructifs : c'est votre opportunité d'aider à améliorer ce cours à l'avenir

Précédemment, dans... ICC-P



- Cryptographie symétrique : une seule clé secrète partagée
- Cryptographie asymétrique :
  - une clé publique pour chiffrer
  - une clé privée pour déchiffrer
- Plus la clé est longue, plus une attaque par force brute est lente
- Révisions

Réseau

Réseau : socket

- Un socket est la prise logicielle — créée par le système d'exploitation — qui permet à un programme d'envoyer ou de recevoir des données sur un réseau.
- Il matérialise une extrémité de communication : d'un côté ton application, de l'autre un autre programme (local ou distant).

Réseau : client et serveur

- **Client**
  - Initie la connexion (appel `connect`)
  - Envoie des requêtes, reçoit des réponses
  - Exemples : navigateur web, appli mobile

- **Serveur**
  - Se lie à une adresse/port (`bind`) puis passe en écoute (`listen`)
  - Accepte les connexions entrantes (`accept`)
  - Traite les requêtes et renvoie les réponses
  - Exemples : serveur HTTP, base de données

Chat : serveur

```python
import socket

HOST = "127.0.0.1" # loopback
PORT = 65432 # port arbitraire (>1024)

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.bind((HOST, PORT))
    s.listen(1)
    print(f"Serveur en écoute sur {HOST}:{PORT}…")
    conn, addr = s.accept()

    with conn:
        print(f"Connecté par {addr}")
        while True:
            data = conn.recv(1024).decode()
            if not data or data.lower() == "bye":
                break
            print(f"[client] {data}")
            msg = input("> ")
            conn.sendall(msg.encode())
            if msg.lower() == "bye":
                break

print("Connexion fermée.")
```

Chat : client

```python
import socket

HOST = "127.0.0.1"
PORT = 65432

with socket.socket(socket.AF_INET, socket.SOCK_STREAM) as s:
    s.connect((HOST, PORT))
    print(f"Connecté au serveur {HOST}:{PORT}. Tapez 'bye' pour quitter.")

    while True:
        msg = input("> ")
        s.sendall(msg.encode())
        if msg.lower() == "bye":
            break
        data = s.recv(1024).decode()
        if not data or data.lower() == "bye":
            break
        print(f"[serveur] {data}")

print("Client terminé.")
```

Résumé Semaine 14 – ICC-P

- Bases réseau : notions de socket et architecture client-serveur
- Démo concrète : mini-chat Python (serveur + client) illustrant

  `bind` → `listen` → `accept/connect` → `send/recv`
- Révisions

rafael.pires@epfl.ch

Merci