第214回 お試しアカウント付き 並列プログラミング講習会 「MPI上級編」受講ガイド

東京大学 情報基盤センター
塙　敏博

内容に関するご質問は
hanawa ＠ cc.u-tokyo.ac.jp
まで、お願いします。

講習会概略

- 開催日：　2023年10月11日（水）　10:00 – 17:30
- 形態：　ZoomおよびSlackを用いたオンライン講習会
- 使用システム：Wisteria/BDEC-01 (Odyssey, Aquarius)
- 講習会プログラム：
  - 10:00 – 11:20　MPI概要、Wisteria/BDEC-01で使えるMPI実装
  - 11:30 - 12:30　ノンブロッキング通信、演習
  - (12:30 – 13:30　お昼休憩)
  - 13:30 – 14:30　派生データ型、MPI-IO、演習
  - 14:40 – 16:10　コミュニケータ、マルチスレッド、演習
  - 16:20 – 17:30　片側通信、演習

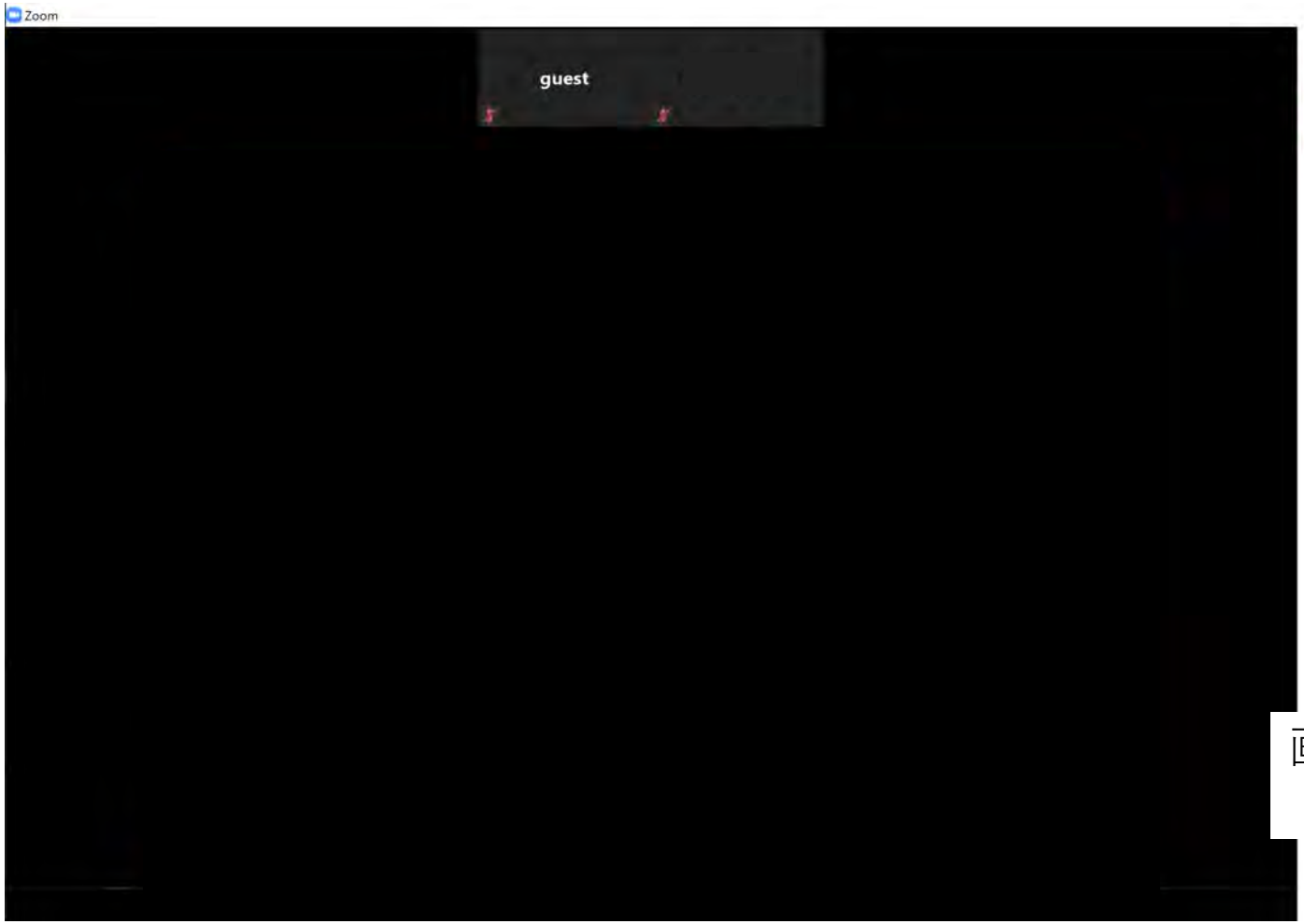
Zoom関連

- 「手をあげる」機能
  - 質問がある際，全体の状況を確認するため使用
- ブレークアウトセッション
  - 画面を共有しながらエラー対応する際に使用
  - （なるべく口頭でのやりとりやSlackで対応する予定）

- https://utelecon.adm.u-tokyo.ac.jp/zoom/how_to_use

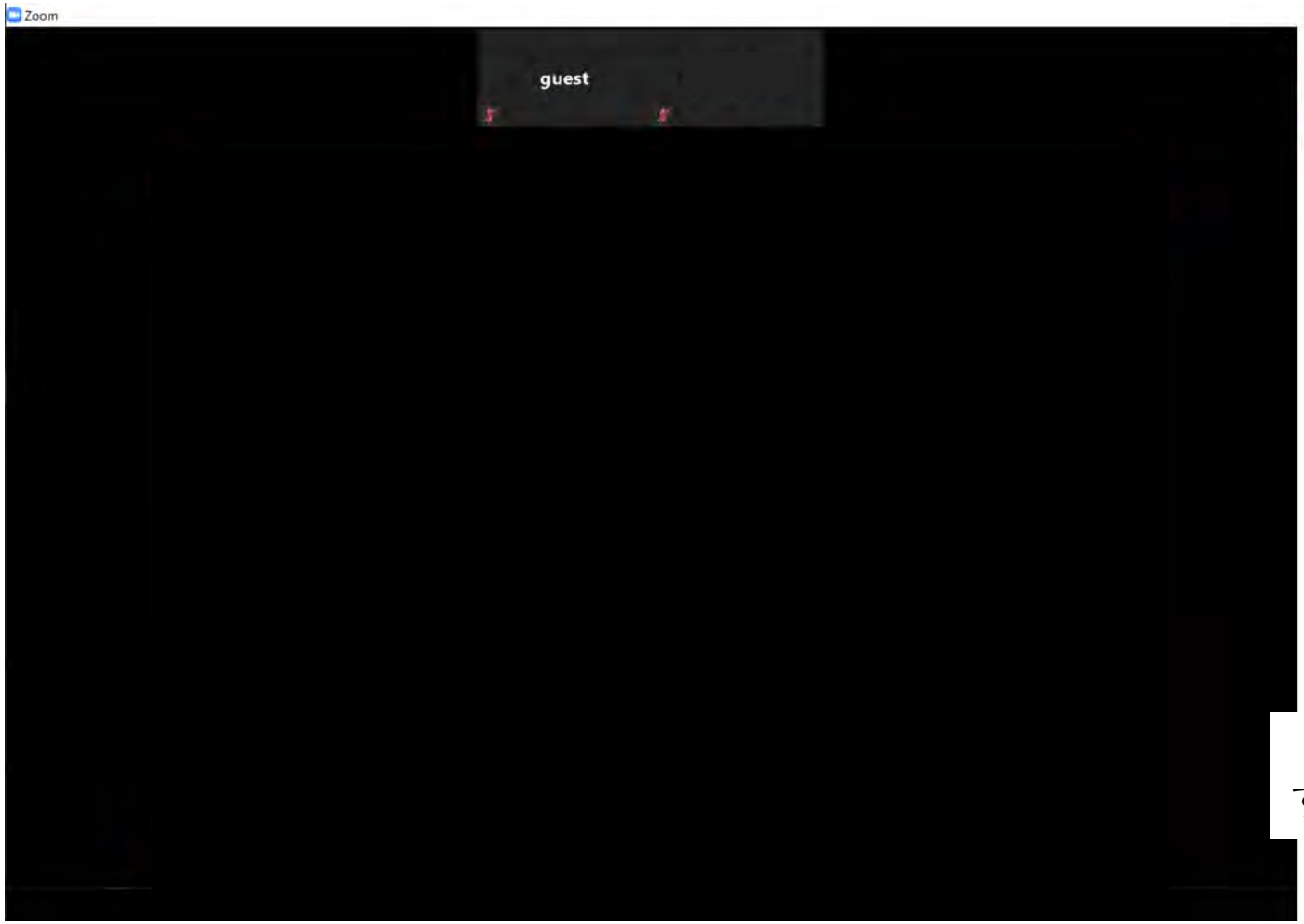
「手を挙げる」機能の使い方（1/3）

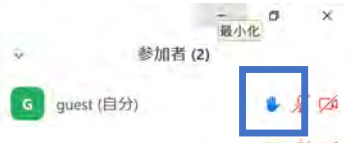
「手を挙げる」機能の使い方（2/3）



画面右下の
「手を挙げる」をクリック

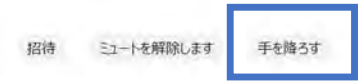
「手を挙げる」機能の使い方（3/3）





自分のところに
手のアイコンが現れる

「手を降ろす」をクリックすると，元に戻る



ブレークアウトセッション（1/4）

- 演習時に使用するかもしれません
- 演習中に「ヘルプを求める」ことができます
  - ホストを招待した後に「画面を共有」することで，皆さんの記述したプログラムを一緒に見ながら問題解決にあたります

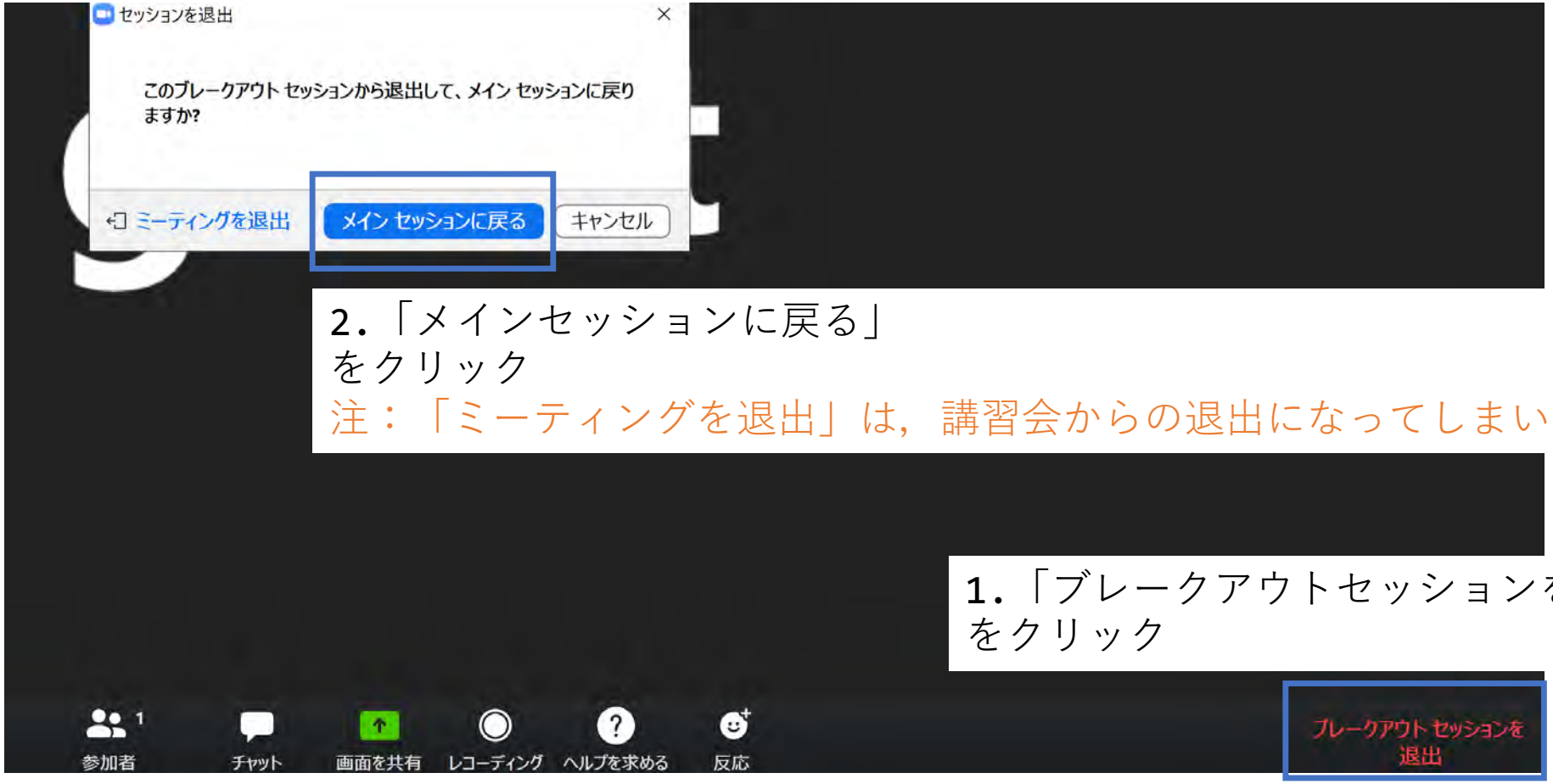
ブレークアウトセッション（2/4）

- この表示が出たら，「ホストを招待」をクリック
- ホストの承認待ちに移行
  - 他の受講者のヘルプ中など，直ちに対応できない場合もあります

ブレークアウトセッション（3/4）

- この状態になると，ホスト（講師）と会話可能
  - マイクの有効化，画面の共有などをしながら相談

ブレークアウトセッション（4/4）



2.「メインセッションに戻る」をクリック

注：「ミーティングを退出」は，講習会からの退出になってしまいます

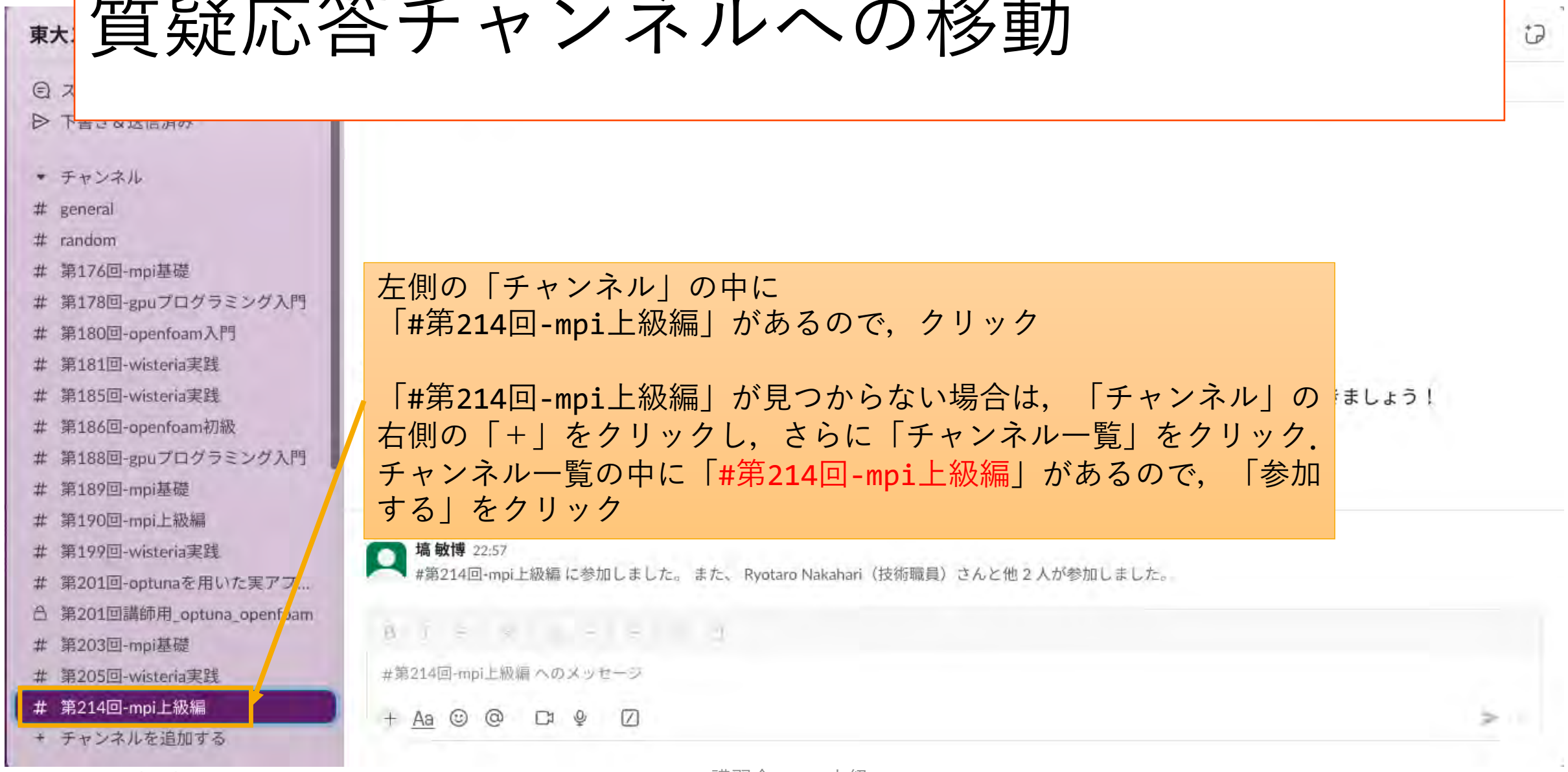
1.「ブレークアウトセッションを退出」をクリック

Slack関連

- ブラウザ上で使う場合には：
  - https://w1590055008-bgo338004.slack.com/archives/C05VA9GEA22
  - 注：ログインには，事前にお配りしたリンクからの登録が必要です
- 質問対応に使用
- コードの貼り付け方
- スレッドの確認方法

- 以下，ブラウザ版で説明しますがアプリ版でも操作は同じです

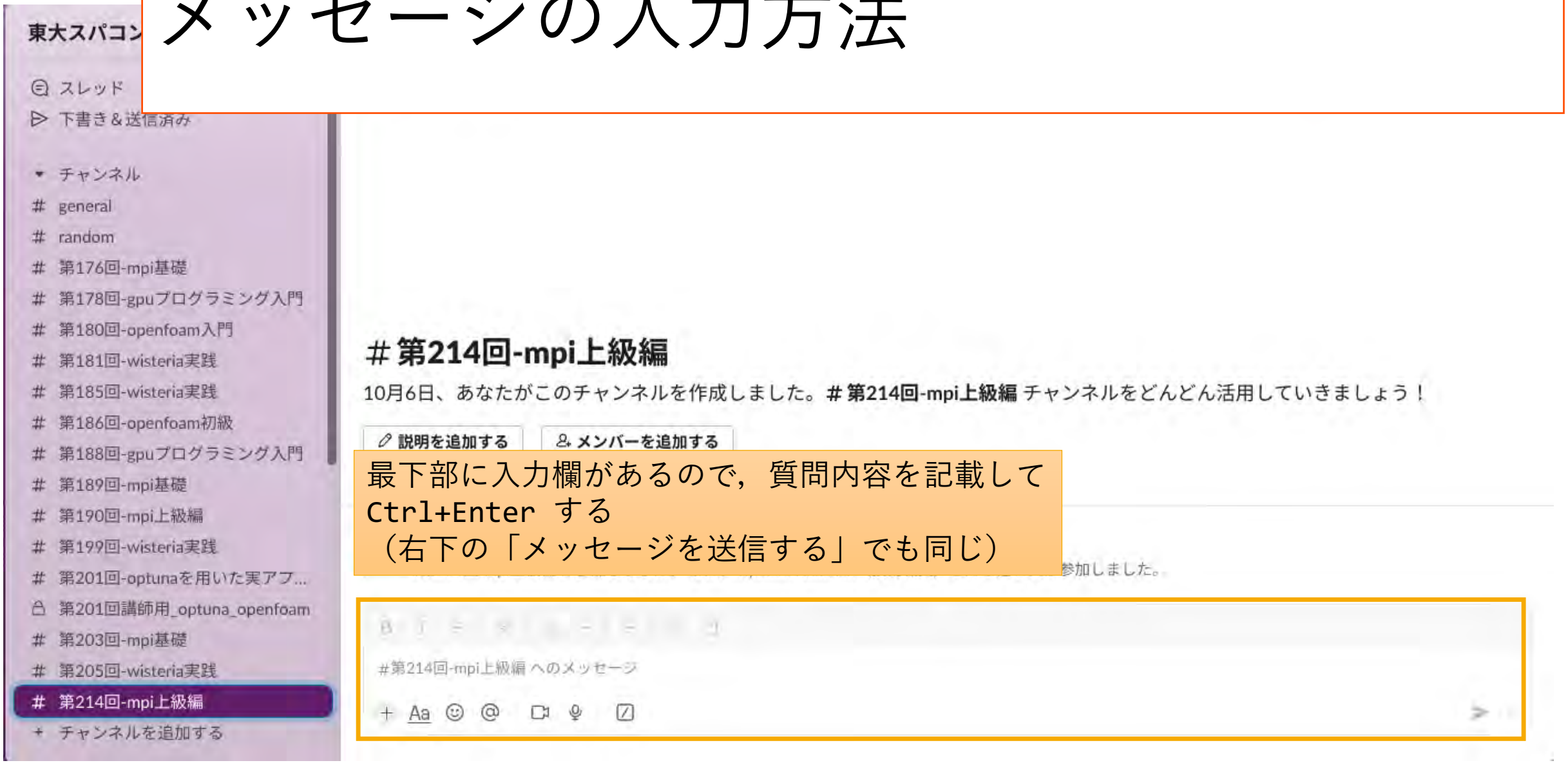
質疑応答チャンネルへの移動



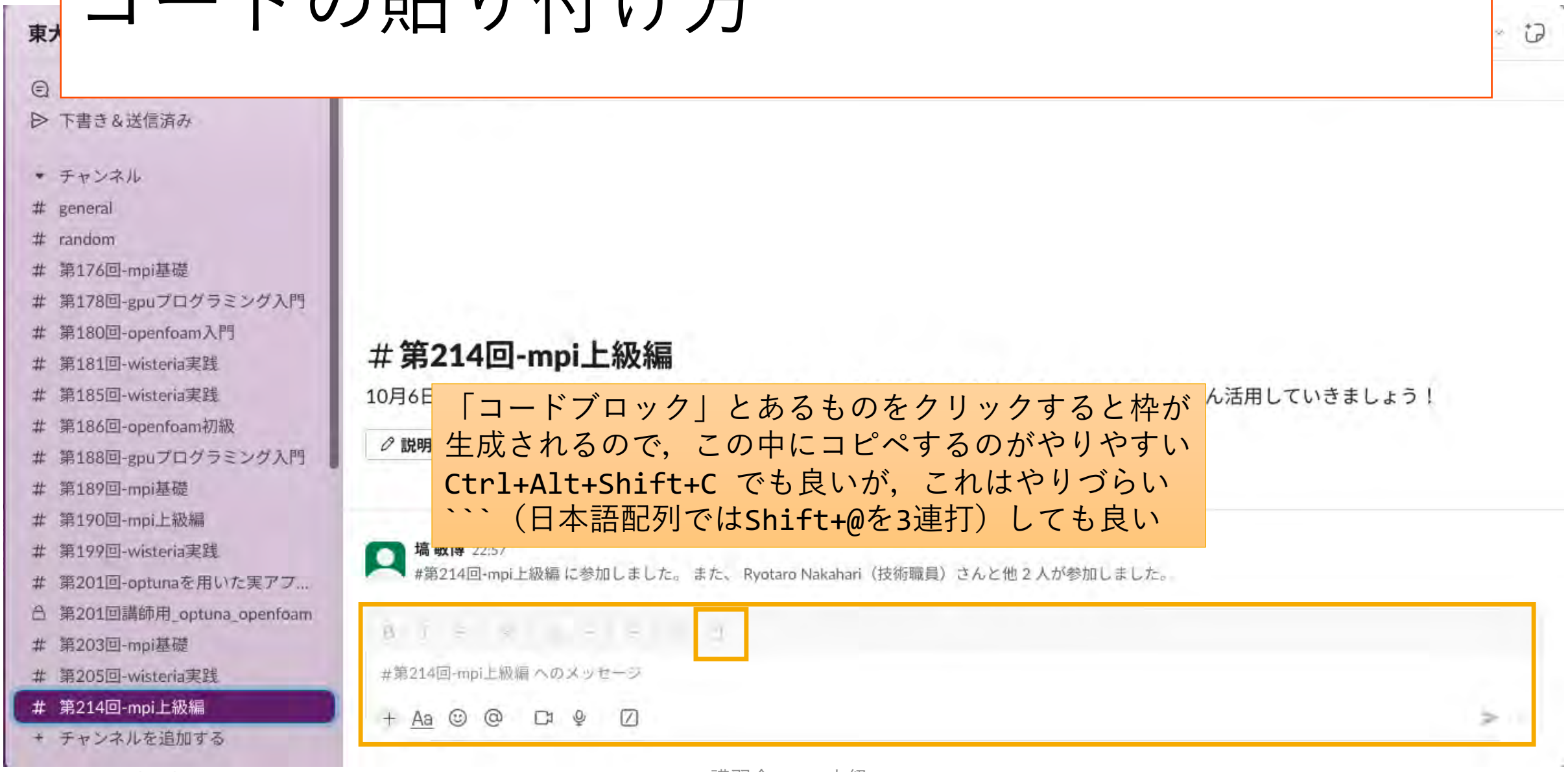
左側の「チャンネル」の中に
「#第214回-mpi上級編」があるので，クリック

「#第214回-mpi上級編」が見つからない場合は，「チャンネル」の右側の「＋」をクリックし，さらに「チャンネル一覧」をクリック．チャンネル一覧の中に「#第214回-mpi上級編」があるので，「参加する」をクリック

メッセージの入力方法

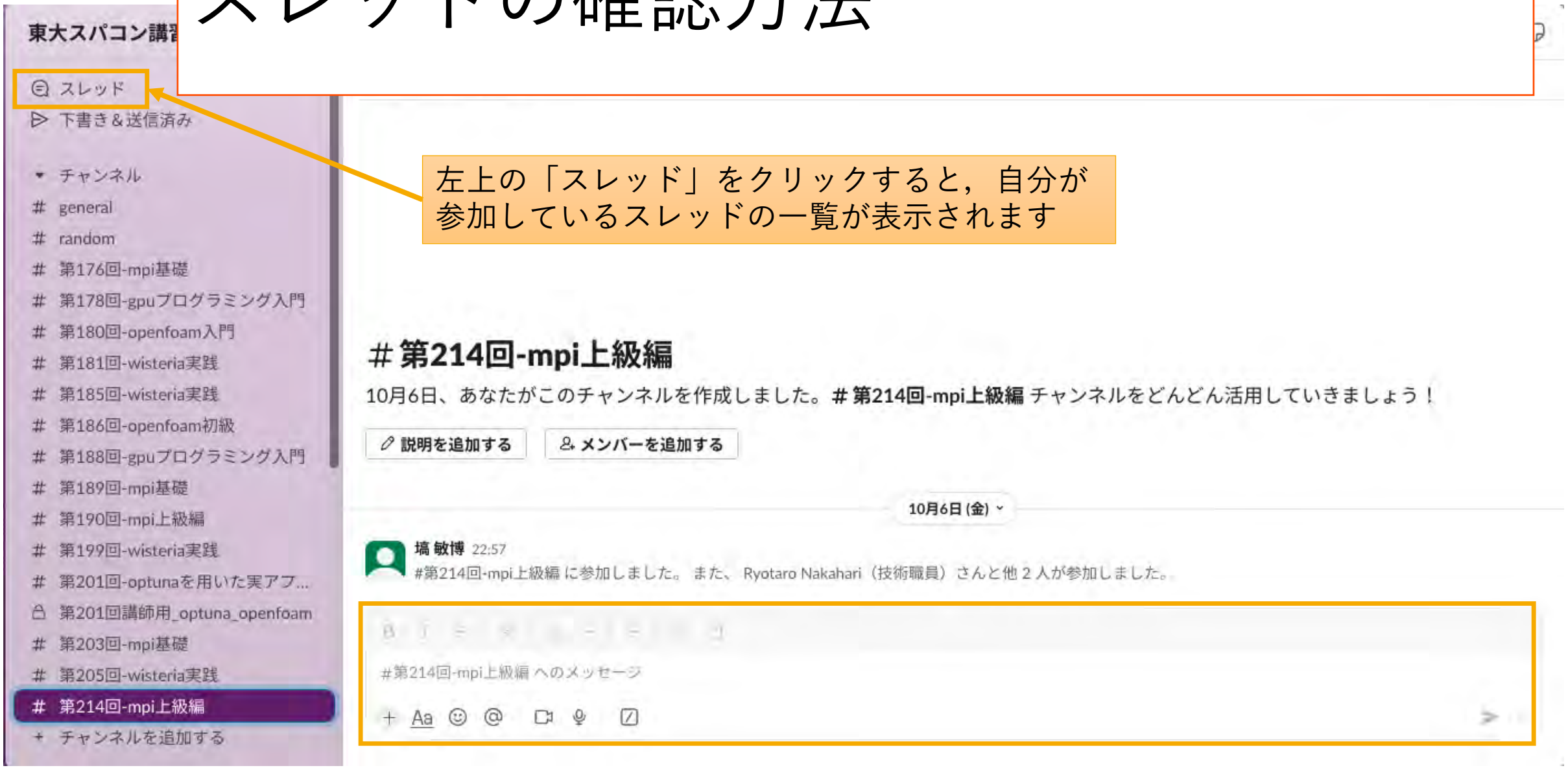

東大スパコン
スレッド
下書き＆送信済み
チャンネル
general
random
第176回-mpi基礎
第178回-gpuプログラミング入門
第180回-openfoam入門
第181回-wisteria実践
第185回-wisteria実践
第186回-openfoam初級
第188回-gpuプログラミング入門
第189回-mpi基礎
第190回-mpi上級編
第199回-wisteria実践
第201回-optunaを用いた実アプ...
第201回講師用_optuna_openfoam
第203回-mpi基礎
第205回-wisteria実践
第214回-mpi上級編
チャンネルを追加する
第214回-mpi上級編
10月6日、あなたがこのチャンネルを作成しました。# 第214回-mpi上級編 チャンネルをどんどん活用していきましょう！
説明を追加する
メンバーを追加する
最下部に入力欄があるので，質問内容を記載して
Ctrl+Enter する
（右下の「メッセージを送信する」でも同じ）
参加しました。
#第214回-mpi上級編 へのメッセージ
Aa

コードの貼り付け方



「コードブロック」とあるものをクリックすると枠が生成されるので，この中にコピペするのがやりやすい
`Ctrl+Alt+Shift+C` でも良いが，これはやりづらい
```（日本語配列では`Shift+@`を3連打）しても良い
```

スレッドの確認方法


東大スパコン講習
スレッド
下書き＆送信済み
チャンネル
general
random
第176回-mpi基礎
第178回-gpuプログラミング入門
第180回-openfoam入門
第181回-wisteria実践
第185回-wisteria実践
第186回-openfoam初級
第188回-gpuプログラミング入門
第189回-mpi基礎
第190回-mpi上級編
第199回-wisteria実践
第201回-optunaを用いた実アプ...
第201回講師用_optuna_openfoam
第203回-mpi基礎
第205回-wisteria実践
第214回-mpi上級編
チャンネルを追加する
左上の「スレッド」をクリックすると，自分が
参加しているスレッドの一覧が表示されます
第214回-mpi上級編
10月6日、あなたがこのチャンネルを作成しました。# 第214回-mpi上級編 チャンネルをどんどん活用していきましょう！
説明を追加する
メンバーを追加する
10月6日 (金)
塙 敏博 22:57
#第214回-mpi上級編 に参加しました。 また、 Ryotaro Nakahari（技術職員）さんと他 2 人が参加しました。
#第214回-mpi上級編 へのメッセージ
Aa

ユーザアカウント

- 使用システム： Wisteria/BDEC-01 (Wisteria)
  - `$ ssh USERNAME@wisteria.cc.u-tokyo.ac.jp`
- 本講習会でのユーザ名
  - 利用者番号： t00471~
  - 利用グループ： gt00
- 利用期限
  - 11/11 9:00まで有効

- 注:本講習会関連の質問はhanawa[at]cc.u-tokyo.ac.jpまで
  - Slackで質問していただいて結構です
  - （講習会アカウントでは）公式の相談対応システムは使わないでください

テストプログラムの概要

- C言語版・Fortran版共通ファイル：
  MPI-advanced.tar.gz
- tarで展開後，物によってはそれぞれの言語用のディレクトリが作られる
  - C/　　：C言語用
  - F/　：Fortran 90用
- 上記ファイルの置き場所：
  /work/gt00/share

サンプルプログラムの取得（1/2）

- 実行してもらうコマンドは $ 以降に青字で記載しています
  - ターミナルへの入力が終わったら 「Enter」 キーを押してください

1. Lustreファイルシステムに移動
   ```
   $ cd /work/gt00/tABCDE          # 下線部は自分のIDに変更
   ```
2. /work/gt00/share/ にあるサンプルファイルをコピー
   ```
   $ cp /work/gt00/share/MPI-advanced.tar.gz .
   ```
   MPI-advanced.tar.gz と .（ドット）の間に半角スペース
3. サンプルファイルを展開
   ```
   $ tar xvfz MPI-advanced.tar.gz
   ```

サンプルプログラムの取得（2/2）

4. MPI-advanced ディレクトリに入る
   ```
   $ cd MPI-advanced
   ```
5. サンプルプログラムのディレクトリがあることを確認
   ```
   $ ls
   ```

Wisteria/BDEC-01でのジョブ実行

- 以下の2通りの実行形態があります

1．バッチジョブ実行

- バッチジョブシステムに処理を依頼して実行
- 実行したい処理をファイル（ジョブスクリプト）で指示
- スパコン環境で一般的
- 大規模実行用
  - Wisteria-Odysseyでは，最大2,304ノード（139264コア），24時間まで

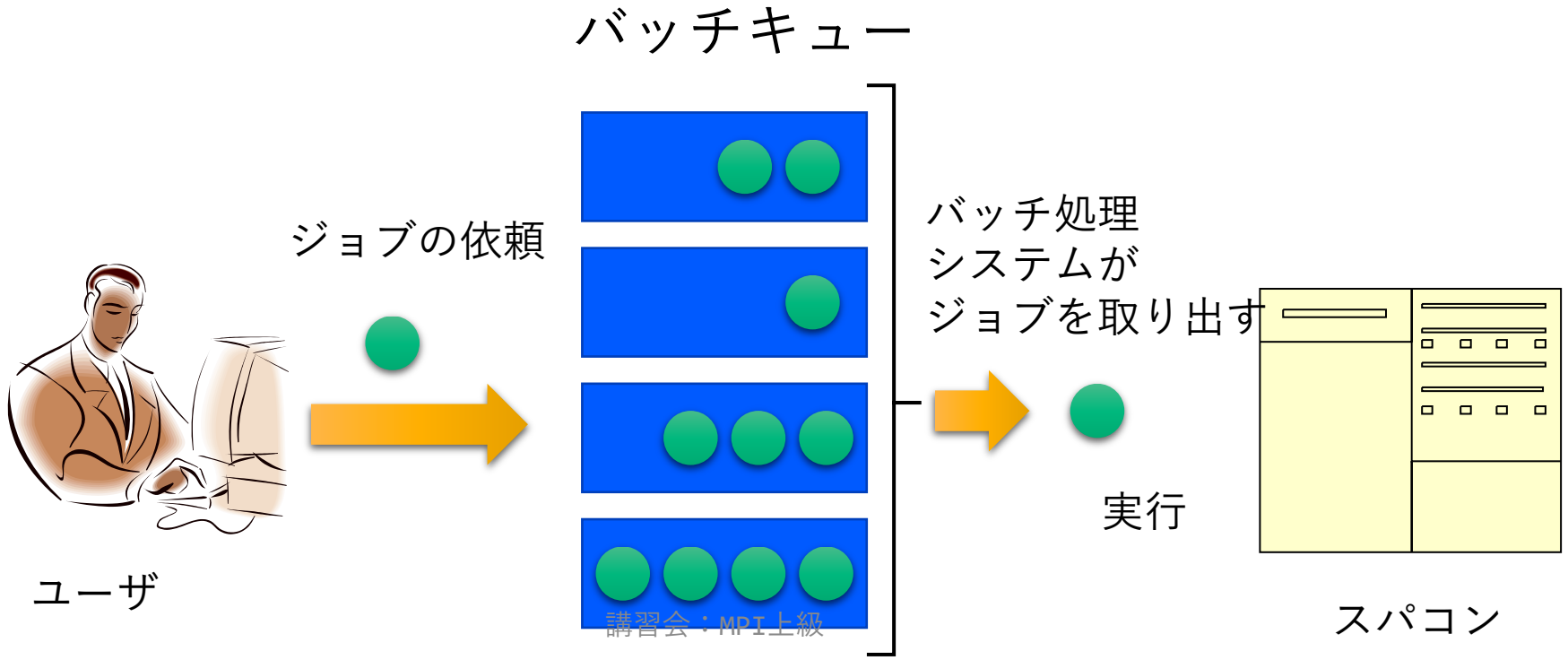
※講習会アカウントでは
バッチジョブ実行のみ，
最大12ノード15分まで

2．インタラクティブジョブ実行

- PCでの実行のように，コマンドを入力して実行
- スパコン環境では一般的ではない
- デバッグ用，大規模実行はできない
  - 1ノード（68コア）：2時間まで
  - 16ノード（1088コア）：10分まで

バッチ処理とは

- スパコン環境では，通常は，インタラクティブ実行（コマンドラインで実行すること）はできません
- ジョブはバッチ処理で実行します



バッチキューの設定方法

- Wisteriaでのバッチ処理は、富士通のバッチシステムで管理
- 主要コマンド：
  - ジョブの投入：pjsub <ジョブスクリプトファイル名>
  - 自分が投入したジョブの状況確認：pjstat
  - 投入ジョブの削除：pjdel <ジョブID>
  - 計算ノードの込み具合を見る: pjstat --rscuse
  - バッチキューの状態を見る：pjstat --rsc
  - バッチキューの詳細構成を見る：pjstat --rsc -x
  - 投げられているジョブ数を見る：pjstat --rsc -b
  - 過去の投入履歴を見る：pjstat -H
  - 同時に投入できる数／実行できる数を見る：pjstat --limit

本お試し講習会でのキュー・グループ名

- 本講習会中のキュー名
  - tutorial-o (Odyssey)
  - 最大15分まで
  - 最大ノード数は12ノード（576コア）まで
  - tutorial-a (Aquarius)
  - 最大15分まで
  - 最大GPU数は4GPUまで
- 本講習会終了後のキュー名
  - lecture-o, lecture-a
  - 利用条件tutorial-o, tutorial-aと同様
- グループ名: gt00

お試し講習会のジョブクラス：バッチジョブ（Odyssey）

キュー名	ノード数	制限時間（Elapsed）	メモリ容量（GB）
lecture-o （教育利用全体で共有）	1 ～ 12	15 分	28
tutorial-o （この講習会の時間中のみ使用可能）	1 ～ 12	15 分	28

ノード当たり 48コア => 最大576コアまで使用可能！

お試し講習会のジョブクラス：バッチジョブ（Aquarius）

キュー名	GPU数	制限時間（Elapsed）	メモリ容量（GB）
lecture-a （教育利用全体で共有）	1,2,4	15 分	56/GPU
tutorial-a （この講習会の時間中のみ使用可能）	1,2,4	15 分	56/GPU

share-aと同様の使い方

ジョブスクリプトの説明（OpenMP/MPIハイブリッド版，Odyssey）

- 内容はC言語，Fortranで共通

```
#!/bin/bash
#PJM -L rscgrp=lecture-o
#PJM -L node=12
#PJM --mpi proc=48
#PJM --omp thread=12
#PJM -L elapse=00:01:00
#PJM -g gt00

module load fj fjmpi
mpiexec  ./hello_omp
```

リソースグループ名：lecture-o

利用ノード数：12ノード使用

MPIプロセス数：48

OpenMPスレッド数：12

実行時間制限：1分

利用グループ名：gt00

MPIジョブを48プロセスで実行

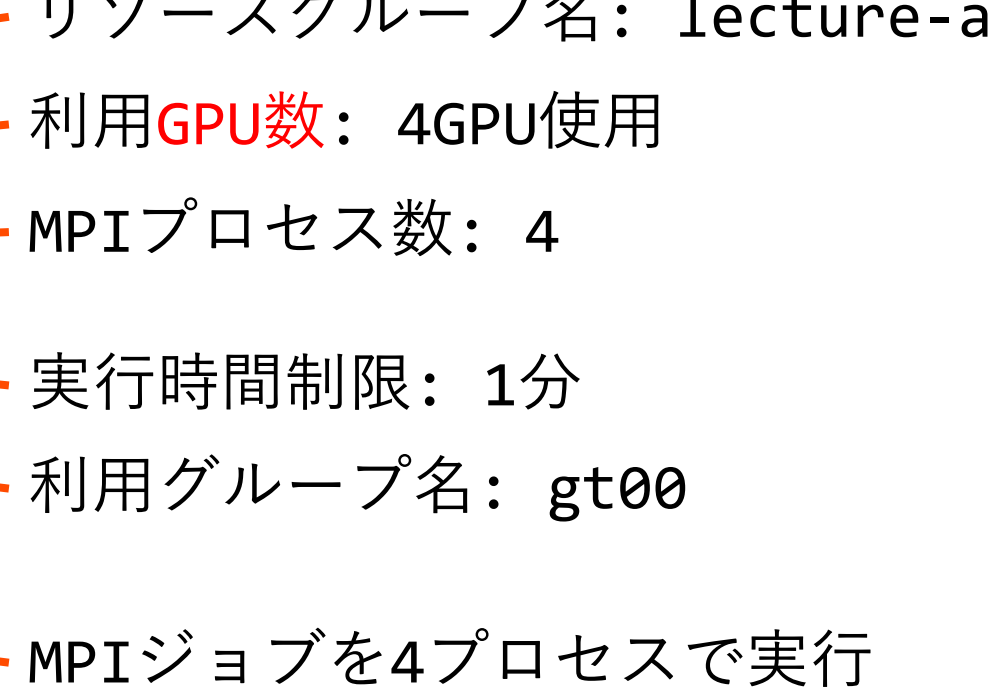
ジョブスクリプトの説明（MPI+OpenACCハイブリッド版，Aquarius）

- 内容はC言語，Fortranで共通

```
#!/bin/bash
#PJM -L rscgrp=lecture-a
#PJM -L gpu=4
#PJM --mpi proc=4
#PJM -L elapse=00:01:00
#PJM -g gt00

module purge
module load nvidia nvmpi

mpiexec -machinefile $PJM_O_NODEINF ¥
-n ${PJM_NUM_PROC} -npernode=4      ¥
 ./hello_omp
```



参考：ジョブクラス：インタラクティヴジョブ（トークン消費無し）

https://www.cc.u-tokyo.ac.jp/supercomputer/wisteria/service/job.php

- Wisteria-O（Odyssey）:シミュレーションノード群

キュー名	ノード数	制限時間（Elapsed）	メモリ容量（GB）
interactive-o (interactive-o_n1) (interactive-o_n12)	 1 2 ～ 12	 30 分 10 分	 28 28

- Wisteria-A（Aquarius）:データ・学習ノード群

キュー名	ノード数	制限時間（Elapsed）	メモリ容量（GB）
interactive-a	1ノード	10 分	448
share-interactive	1GPU	10 分	56

参考：インタラクティブ実行のやり方（本講習会では使えません）

Odyssey

- 1ノード実行

```
$ pjsub --interact -g グループ名 -L rg=interactive-o,elapse=01:00
```

- 12ノード実行

```
$ pjsub --interact -g グループ名 -L rg=interactive-o,node=12,elapse=01:00
```

Aquarius

- 1GPU実行

```
$ pjsub --interact -g グループ名 -L rg=share-interactive,elapse=01:00
```

- 1ノード(8GPU)実行

```
$ pjsub --interact -g グループ名 -L rg=interactive-a,elapse=01:00
```

※インタラクティブ用のノードがすべて使われている場合、資源が空くまでログインできません。
※このアカウントでは使えません。

参考：ジョブクラス：バッチジョブ（Odyssey）

https://www.cc.u-tokyo.ac.jp/supercomputer/wisteria/service/job.php

キュー名	ノード数	制限時間（Elapsed）	メモリ容量（GB）
debug-o	1 ～ 144	30 分	28
short-o	1 ～ 72	8 時間	28
regular-o (small-o) (medium-o) (large-o) (x-large-o)	 1 ～ 144 145 ～ 576 577 ～ 1152 1153 ～ 2304	 48 時間 〃 〃 24 時間	 28 〃 〃 〃
priority-o （優先実行，トークン消費量 1.5倍）	1 ～ 288	48 時間	28

参考：ジョブクラス：バッチジョブ（Aquarius）

https://www.cc.u-tokyo.ac.jp/supercomputer/wisteria/service/job.php

キュー名	ノード数 GPU数	制限時間 （Elapsed）	メモリ容量 （GB）
debug-a	1 ノード	30 分	448
short-a	1 ～ 2 ノード	2 時間	448
regular-a 　(small-a) 　(medium-a) 　(large-a)	 1 ～ 2 ノード 3 ～ 4 ノード 5 ～ 8 ノード	 48 時間 〃 24 時間	 448 〃 〃
share-debug	1, 2, 4 GPU	30 分	56
share-short	1, 2, 4 GPU	2 時間	56
share 　(share-1) 　(share-2) 　(share-4)	 1 GPU 2 GPU 4 GPU	 48 時間 〃 24 時間	 56 〃 〃

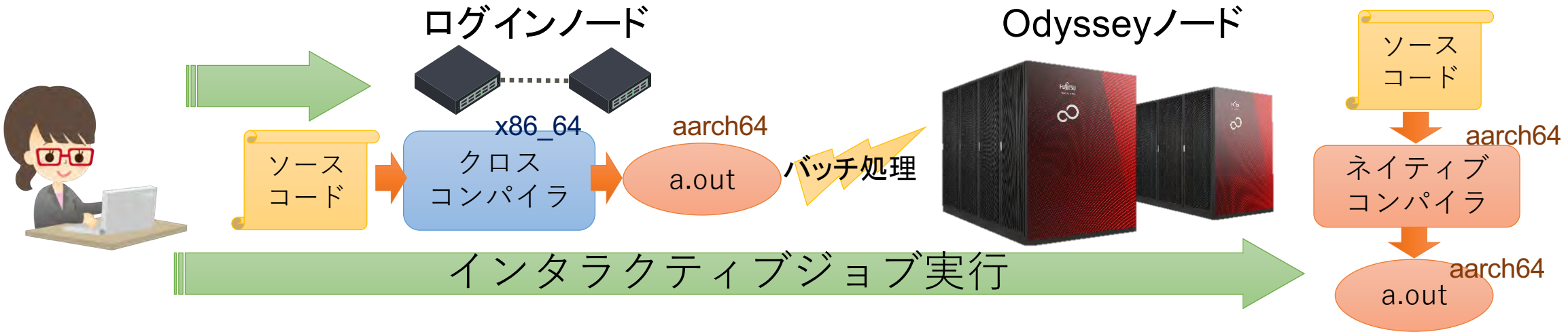
参考：ジョブクラス：プリポストサービス
ログインノードと同じアーキテクチャのノードを使用

https://www.cc.u-tokyo.ac.jp/supercomputer/wisteria/service/job.php

キュー名	ノード数	制限時間（Elapsed）	メモリ容量（GB）
prepost（予約無し）	1	6 時間	340
prepost1_n1 ～ prepost4_n1	1	1 ～ 6 時間	340
prepost1_n4	1 ～ 4	1 ～ 6 時間	340
prepost1_n8	1 ～ 8	1 ～ 6 時間	340

コンパイラの種類と実行(Odyssey)

- ログインノードとOdysseyの計算ノードとで、CPUの命令セットが大きく異なる
  - ログインノード：命令セットアーキテクチャ Intel CascadeLake＋AVX512, **x86_64**
  - Odyssey計算ノード： Fujitsu A64FX，命令セットアーキテクチャ ARM v8.2 + SVE, **aarch64**



- 富士通製コンパイラ
  - `module load odyssey` で使用可能になる
  - ネイティブコンパイラ： ＜コンパイラの種類名＞
  - クロスコンパイラ： ＜コンパイラの種類名＞px
  - MPI: mpi+コンパイラ名 (例：mpifccpx)

言語	ネイティブコンパイラ	クロスコンパイラ
C	fcc	fccpx
C++	FCC	FCCpx
Fortran	frt	frtpx

プログラムの実行

- ここでジョブスクリプト名は run.sh とします
- 配布したサンプルではキュー名が“lecture-o”になっているので，これを“tutorial-o”に変更してください

```
$ emacs -nw run.sh    # emacs で編集する場合
$ vim run.sh          # vim   で編集する場合
$ nano run.sh         # nano  で編集する場合
```

- ジョブを投入

```
$ pjsub run.sh
```