<u>การนำ source code ภาษาซี ขึ้นไปยังเครื่อง frontend</u>

1. Download และ ติดตั้ง WinSCP จาก: https://winscp.net/download/WinSCP-5.15.9-Setup.exe
2. หลังจากเปิดโปรแกรมให้เลือก New Site จากนั้นแก้ไขข้อมูลดังนี้

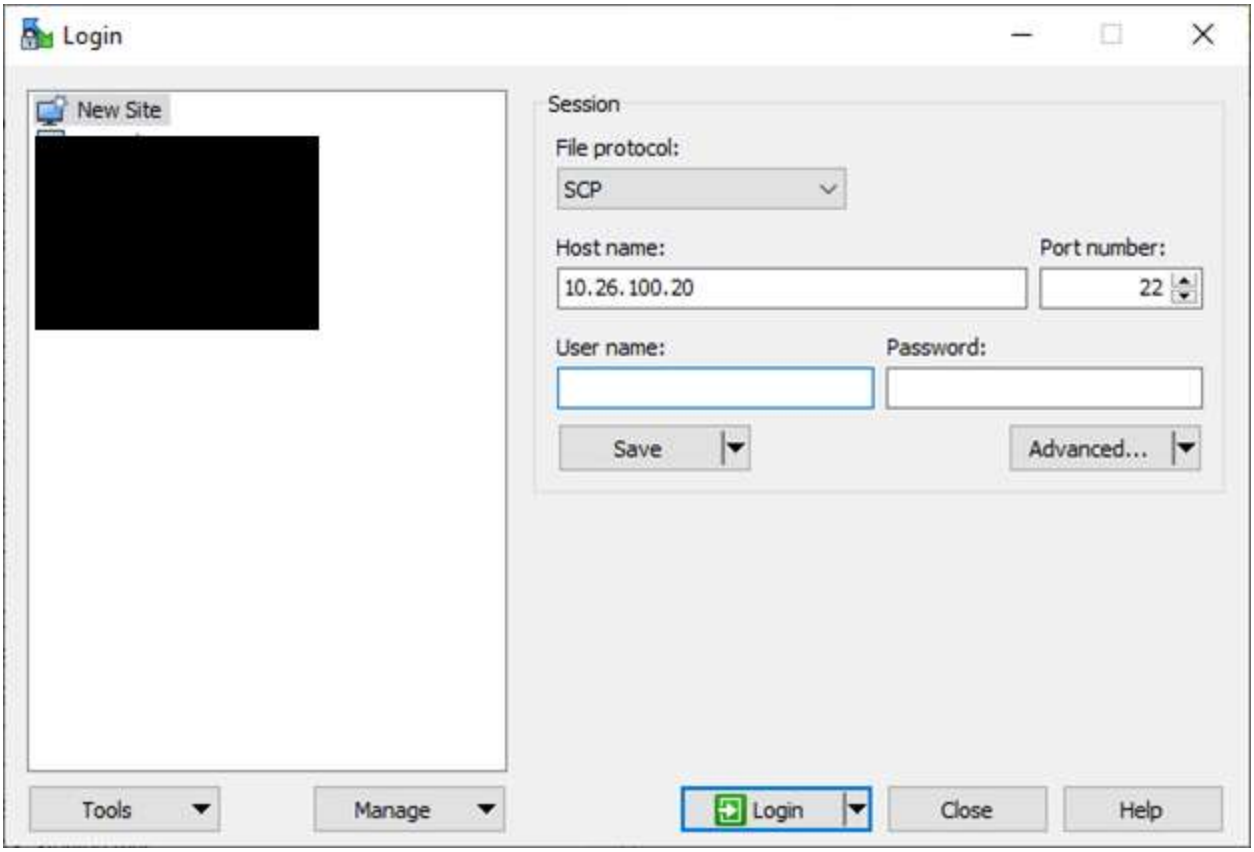
   File Protocol: SCP

   Host name: cluster.kscc.innosoft.kmutt.ac.th

   Port number: 2020

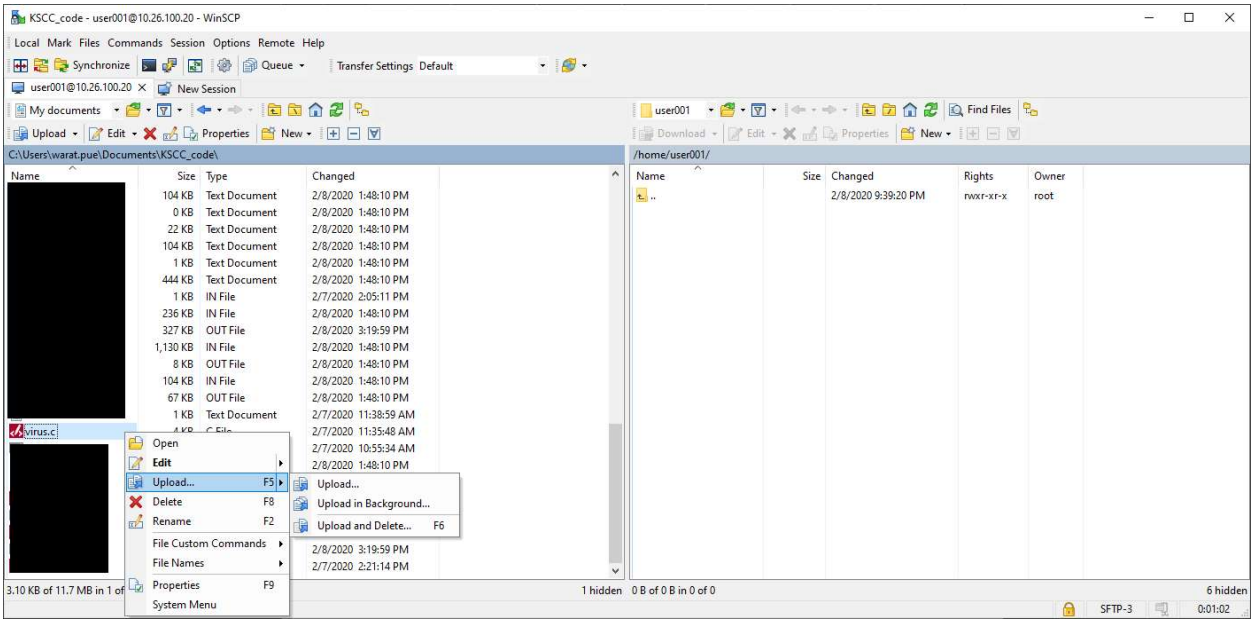
   Username: <ตามที่ได้แจกไว้ให้>

   Password: <ตามที่ได้ทำการแก้ไข>



3. หลังจากกรอกข้อมูลแล้วสามารถ Save เพื่อความสะดวกในการเชื่อมต่อครั้งถัดไป จากนั้นกด **Login** เพื่อทำการเชื่อมต่อ

4. เมื่อทำการเชื่อมต่อเสร็จสิ้นจะพบกับหน้าจอในลักษณะด้านล่าง



5. Click ขวา file ที่ต้องการ ย้ายไปยังเครื่อง frontend จากนั้นเลือก upload

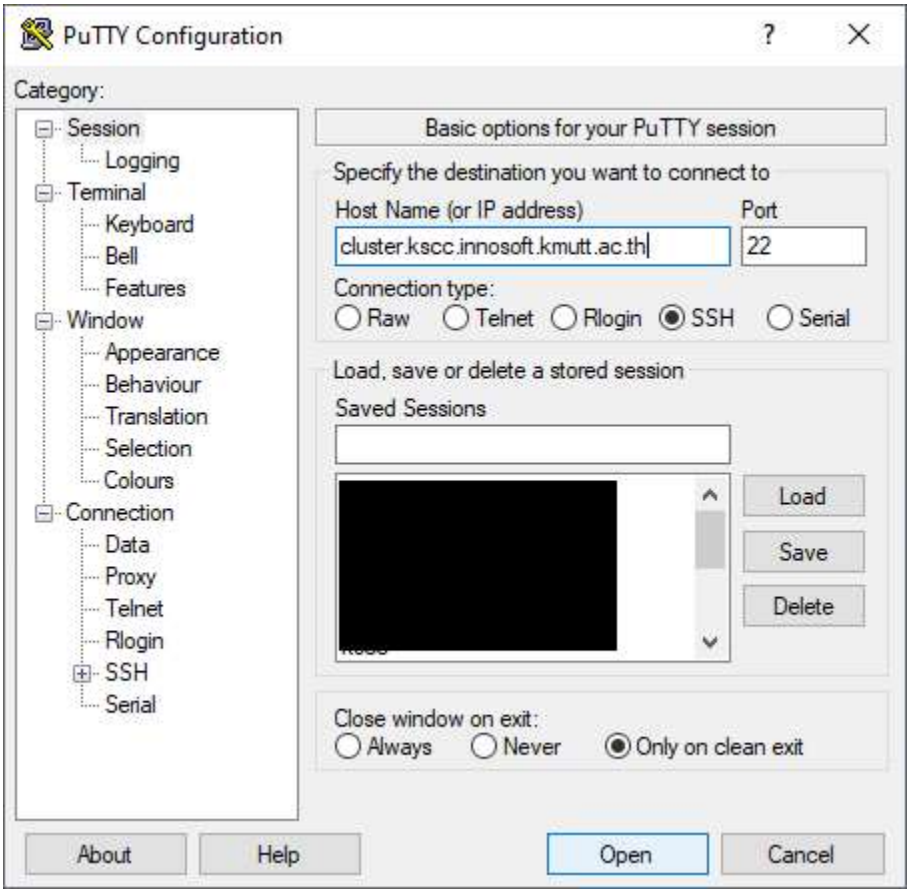
การ Secure Shell เข้าไปยังเครื่อง server เพื่อ การ compile และ รัน

1. Download และติดตั้ง putty จาก https://the.earth.li/~sgtatham/putty/latest/w32/putty-0.73-installer.msi
2. หลังจากเปิดโปรแกรม ให้กรอกค่าดังนี้ จากนั้นกด Open

   Hostname: cluster.kscc.innosoft.kmutt.ac.th

   Port: 2020

   Connection type: SSH



3. กรอก username และ password ที่ได้ไปเพื่อทำการเชื่อมต่อ

การสร้าง job file และ submit

การส่งงานเข้าประมวลผล ผ่าน SGE นั้นประกอบด้วย 2 ส่วน คือ job file และคำสั่ง qsub

Job file เราจะสร้างไว้ใน directory ที่ต้องการ run งาน โดยมีส่วนประกอบที่สำคัญดังนี้

```
#!/bin/bash
#$ -cwd
#$ -N [your_job_name]
#$ -q [queue_name.q]

[your_program] [argv1] [argv2] [argv3] ...
```

โดยแต่ละบรรทัดมีความหมายดังนี้

- -cwd หมายถึง ให้ใช้ directory ปัจจุบันนี้ เป็น working directory
- -N [your_job_name] สำหรับระบุ job name เพื่อง่ายสำหรับการ monitoring ต่อไป
- -q [queue_name.q] สำหรับเลือก queue ที่จะส่งงานเข้าไปรันสำหรับ การทดลองบนเครื่อง cluster ให้ใช้ kscc.q

และสำหรับ ผู้ที่ต้องการ run แบบ Parallel ด้วย MPI สามารถเลือกจำนวนหน่วยประมวลผลที่ต้องการได้ โดยเพิ่มบรรทัด -pe และตัวแปร $NSLOTS เพื่อ สร้าง Parallel Environment และระบุจำหน่วยหน่ายประมวลผลที่ต้องการ โดยผู้ใช้แต่ละคนจะใช้หน่วยประมวลผลรวมทั้งหมดทุก Job ได้สูงสุดไม่เกิน 12 โดย SGE จะจัดการเลือกเครื่องที่ว่างสำหรับ run งาน ให้โดยอัตโนมัติ โดยให้ slots ในการประมวลผลเท่ากับจำนวนที่ผู้ใช้ระบุ โดยไม่ต้องระบุ machine file เพิ่มเติม ดังตัวอย่างด้านล่าง

```
#!/bin/bash
#$ -cwd
#$ -N [your_job_name]
#$ -q [queue_name]
#$ -pe mpi [number_of_slots]

mpirun -np $NSLOTS [your_program] [argv1] [argv2] [argv3] ...
```

สำหรับผู้ที่ต้องการ run ด้วย OpenMP จะมีลักษณะ Job File ดังนี้

```
#!/bin/bash
#$ -cwd
#$ -N [your_job_name]
#$ -q [queue_name]
#$ -pe mpi [number_of_slots]

export OMP_NUM_THREADS=$NSLOTS && [your_program] [argv1] [argv2]
```

และ สำหรับผู้ที่ต้องการ run ทั้ง MPI และ OpenMP ผสมกัน

```
#!/bin/bash
#$ -cwd
#$ -N [your_job_name]
#$ -q [queue_name]
#$ -pe mpi [total number_of_slots]

export OMP_NUM_THREADS=[processor per computenode] && mpirun -n [number of compute
node] -x OMP_NUM_SLOTS -pernode  [your_program]
```

การ Submit Job

เมื่อสร้าง job file เสร็จแล้วเราก็จะทำการ submit job เข้าไป queue ด้วยคำสั่ง qsub

```
qsub [job_file]
```

SGE ก็จะแจ้ง Job ID ซึ่งเป็นหมายเลขอ้างอิง Job ให้ทราบ สมมติให้)Job ID เป็น 1234)

```
Your job 1234 ("job_name") has been submitted.
```

เมื่อ Job เสร็จเรียบร้อยภายใน working directory ก็จะปรากฏ 2 ไฟล์

```
[your_job_name].e1234
[your_job_name].o1234
```

ซึ่ง .e จะเป็นข้อมูลจาก stderr ของโปรแกรม และ .o จะเป็นเป็นข้อมูลจาก stdout และหากมีการสร้าง Parallel Environment ก็จะมีไฟล์ .pe และ .po เพิ่มเข้ามาด้วย

ตัวอย่าง

ตัวอย่างนี้ เพื่อให้เห็นภาพและทดลองทำตามได้โดยง่าย โดยใช้โปรแกรมคูณ matrix ด้วย MPI

```
me@hpc example]$ ls -l
total 32
-rw-rw-r-- 1 me me  4992 Oct 28 14:28 matrix.tar.gz
-rw-rw-r-- 1 me me    95 Oct 28 14:05 m.job
-rwxrwxr-x 1 me me 10230 Oct 28 13:44 mmult
-rw-rw-r-- 1 me me  2340 Oct 28 13:44 mmult.c
```

เราต้องการจะส่งไฟล์โปรแกรมของเราที่มีชื่อ ว่า mmult เข้า queue เพื่อประมวลผล เราต้องมี job file ซึ่งในตัวอย่างนี้ job file ของเรามีชื่อว่า m.job ซึ่งภายในไฟล์ก็จะมีเนื้อหาดังนี้

```
#!/bin/bash
#$ -cwd
#$ -N matrix_mult
#$ -q kscc.q
#$ -pe mpi 2

mpirun -np $NSLOTS ./mmult
```

บรรทัดที่ขึ้นต้นด้วย #$ ทั้ง 4 บรรทัด จะเป็น option สำหรับบอกรายละเอียดให้กับ SGE ดังนี้

- -cwd เพื่อบอก current working directory
- -N matrix_mult สำหรับตั้งชื่อให้ Job โดยในที่นี้ ให้ชื่อว่า matrix_mult
- -q kscc.q สำหรับเลือกให้ Job ของเรานั้นเข้า queue ชื่อ kscc.q
- -pe mpi สำหรับบอกว่า Job ของเราต้องการ run บน Parallel Environment ที่ชื่อว่า mpi โดยใช้ 2 slots

และในบรรทัดสุดท้าย เป็นการสั่ง run โปรแกรมโดยสร้าง process ให้มีจำนวนเท่ากับจำนวน slots ที่ร้องขอด้วยตัวแปร $NSLOTS

เราจะทำการ submit job เข้าไปใน queue ด้วยคำสั่ง qsub

```
[me@hpc example]$ qsub m.job
Your job 531 ("matrix_mult") has been submitted
```

หลังจาก submit job เรียบร้อย เราก็จะทำการติดตาม job ของเราด้วย qstat

```
[me@hpc example]$ qstat
job-ID  prior   name       user         state submit/start at     queue
slots ja-task-ID
-----------------------------------------------------------------------------------------------------
---------------
    531 0.50500 matrix mul me           r     10/28/2014 15:05:22 kscc.q@compute-0-7.local
2
```

เมื่อ Job เสร็จเรียบร้อย เราก็จะพบไฟล์ 4 ไฟล์ใน example directory

```
[me@hpc example]$ ls -l
total 40
-rw-r--r-- 1 me me     0 Oct 28 15:05 matrix_mult.e531
-rw-r--r-- 1 me me  1014 Oct 28 15:05 matrix_mult.o531
-rw-r--r-- 1 me me     0 Oct 28 15:05 matrix_mult.pe531
-rw-r--r-- 1 me me     0 Oct 28 15:05 matrix_mult.po531
-rw-rw-r-- 1 me me  4992 Oct 28 14:28 matrix.tar.gz
-rw-rw-r-- 1 me me    95 Oct 28 14:05 m.job
-rwxrwxr-x 1 me me 10230 Oct 28 13:44 mmult
-rw-rw-r-- 1 me me  2340 Oct 28 13:44 mmult.c
```

ซึ่งข้อมูลที่ออกจากโปรแกรมทาง stderr จะอยู่ในไฟล์ matrix_mult.e531 และ stdout จะอยู่ใน matrix_mult.o531 ซึ่งเมื่อเปิดไฟล์ .o ก็จะได้ผลลัพท์จากการประมวลผลนั่นเอง

Monitoring (qstat)

qstat ใช้สำหรับติดตามว่า job ทั้งหมดของเรานั้น อยู่ในสถานะใดแล้ว

```
qstat
```

เมื่อเรียกคำสั่ง qstat ก็จะปรากฏ list ของ job ที่เราส่งเข้า queue ไว้

```
job-ID  prior   name       user         state submit/start at
queue                          slots ja-task-ID
-----------------------------------------------------------------------
-----------------------------------------
   1234 0.55500 my_job1    my_username  r     10/01/2014 09:48:04
all.q@compute-0-5.local           1
   1245 0.55500 my_job2    my_username  qw    10/01/2014 10:50:19
all.q@compute-0-9.local           8
```

คำสั่ง qstat นั้น แสดงสถานะของงานได้จาก state ที่แตกต่างกัน ดังนี้

- **qw** รอคิวเพื่อเริ่มประมวลผล
- **t** กำลังส่งข้อมูลไปยัง compute node
- **r** กำลังประมวลผล
- **h** job held back โดยผู้ใช้
- **E** error. (กรุณาติดต่อผู้ดูแลระบบ)

Deleting (qdel)

สำหรับยกเลิกการทำงาน โดยเราจะเรียกคำสั่ง qdel พร้อมระบุ job ID ที่ต้องการยกเลิก (สมมติให้ Job ID ที่ต้องการคือ 1234) เช่น

```
qdel 1234
```

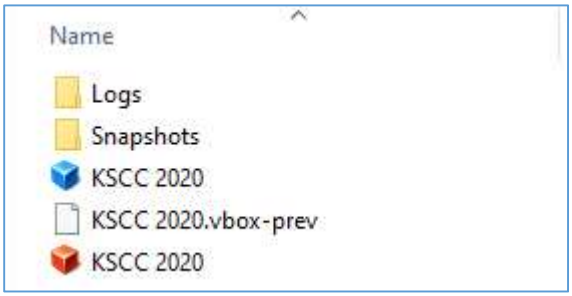
พร้อมข้อความแจ้งกลับมาว่า

```
[username] has registered the job 1234 for deletion
```

ขั้นตอนการติดตั้งและใช้งาน Virtual Machine สำหรับฝึกหัด OpenMP และ MPI เพื่อทดสอบบนเครื่องส่วนตัว

Prerequisites (เครื่องคอมพิวเตอร์ต้องมี RAM มากกว่า 4 GB และ Storage มากกว่า 25 GB)

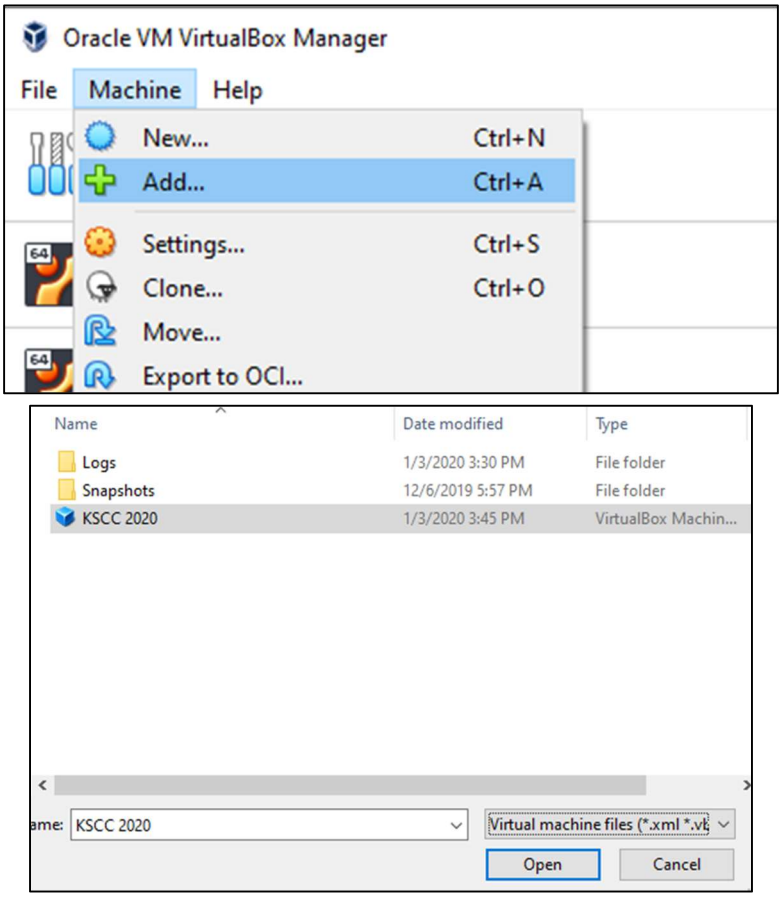
1. Download Zip file จาก URL https://drive.google.com/file/d/15f54HGZD4xIHHBgyWilCvsjJ-17UwDGm
2. Extract Zip File ออกมาจะได้ folder VM-KSCC 2020 ซึ่งจะมี file KSCC 2020 อยู่ด้านใน ดังรูป สำหรับตัว Virtual Machine จะเป็นระบบปฏิบัติการ CentOS 7 ซึ่งได้มีการติดตั้ง Library ที่จำเป็น พร้อมทั้ง Visual Studio Code



3. Download Oracle Virtual Box จาก website: https://www.virtualbox.org/ และดำเนินการติดตั้ง

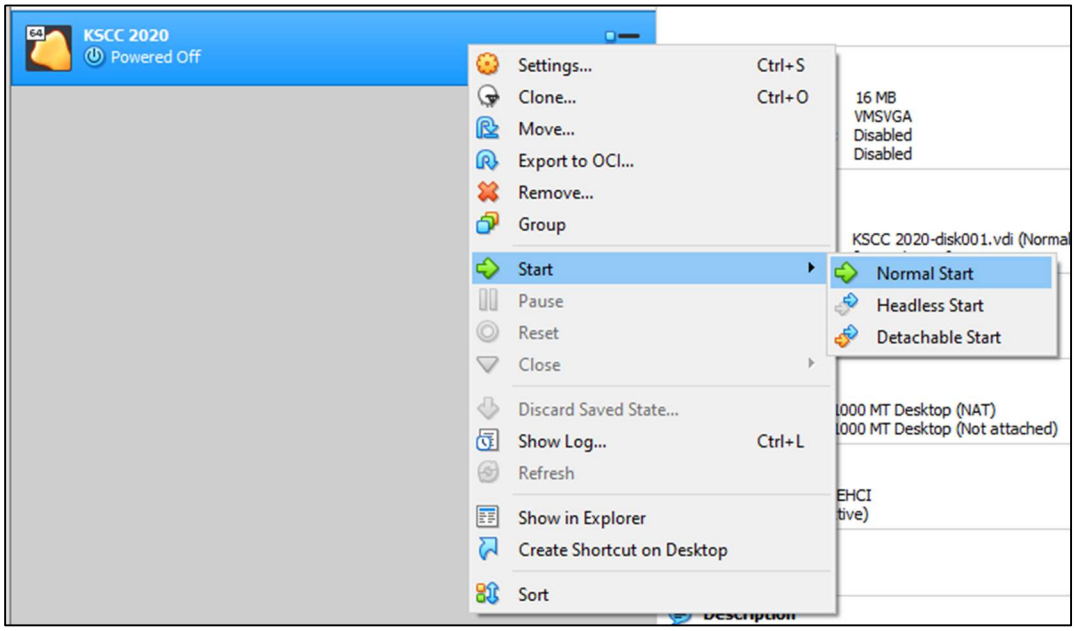
Deploy ova file into Virtual Box

1. เปิด โปรแกรม Virtual Box ไปทีเมนู Machine > Add จากนั้นเลือก KSCC 2020 ที่ได้ extract ไว้ จากนั้นกด open



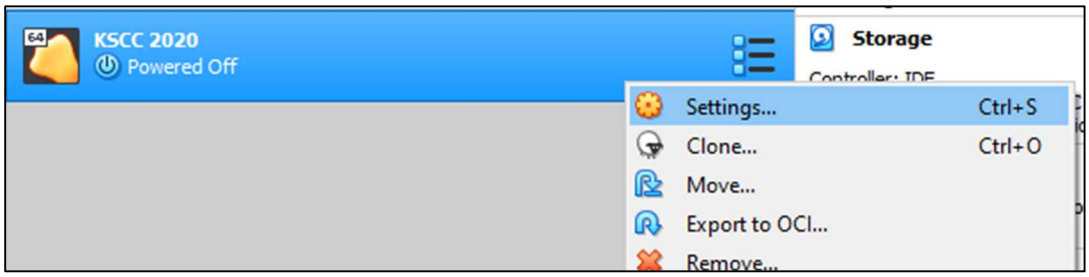
2. หลังจาก Add เสร็จสิ้นจะได้ Virtual Machine ใน VirtualBox ดังรูป

3. ดำเนินการ Start Virtual Machine ด้วยการ Right Click ที่ KSCC 2020 > Start > Normal Start ดังรูป

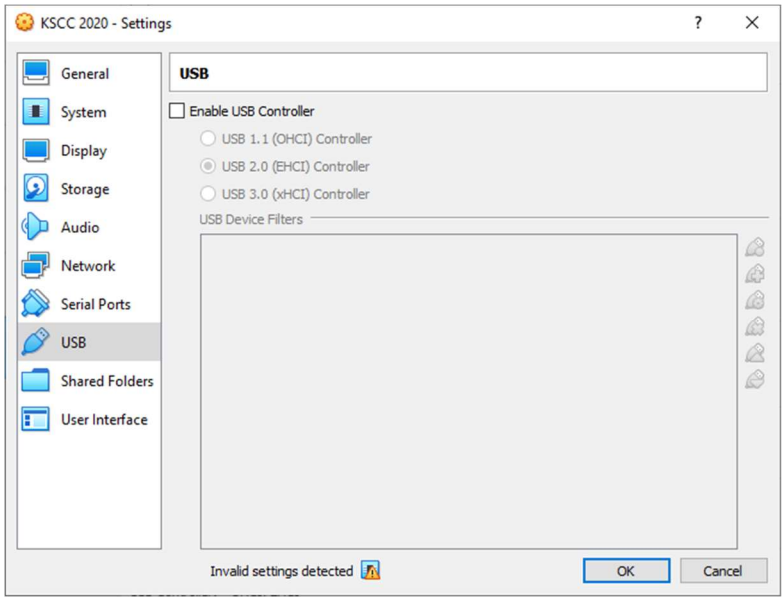


** หมายเหตุ หากทำการ Start Virtual Machine แล้วพบข้อความดังรูปด้านล่างให้ทำตามข้อ 3.1 – 3.3

3.1 click ขวาที่ Virtual Machine > Setting

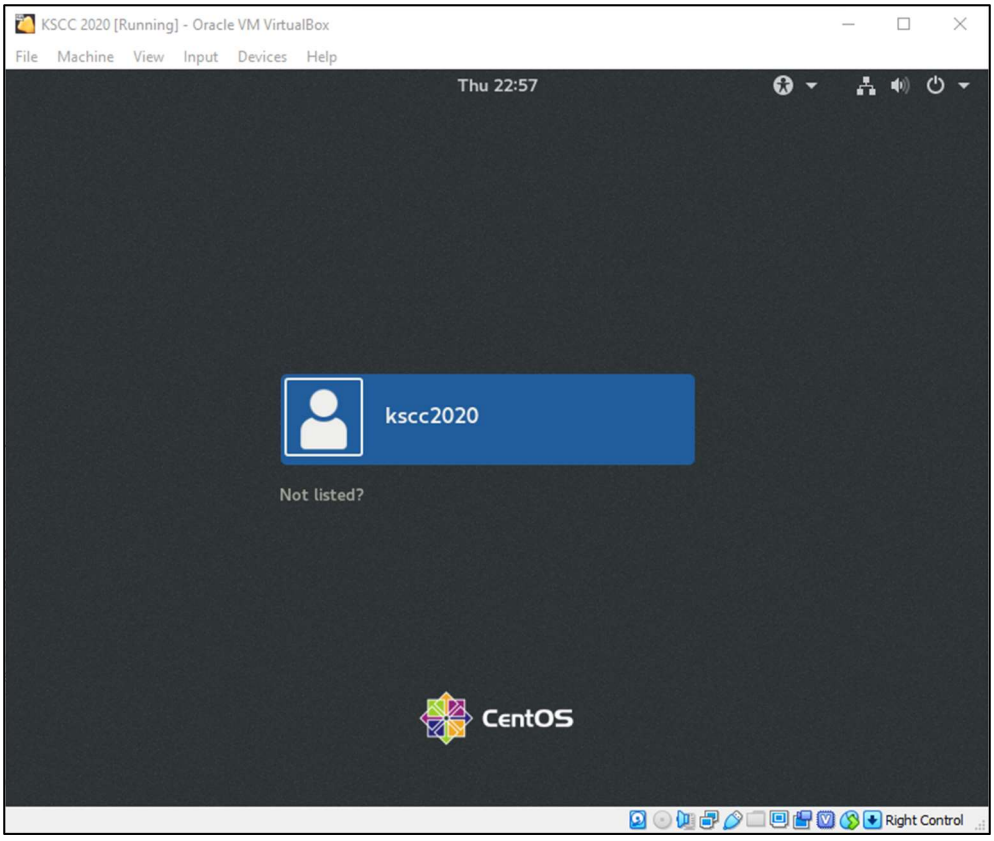


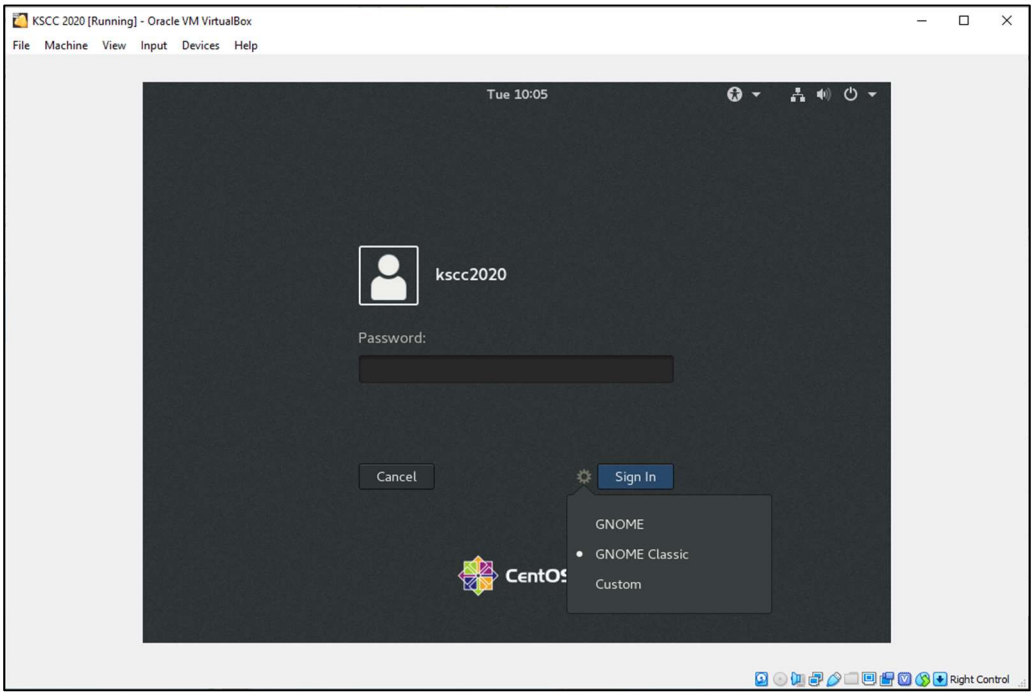
3.2 ไปที่ Tab USB หลังจากนั้น check out กล่องด้านหน้า “Enable USB Controller” ออก ดังรูป



3.3 หลังจากนั้นกด OK และทำการ Start Virtual Machine ใหม่อีกครั้ง

4. หลังจาก Start จะมีหน้าต่างของ Virtual Machine ขึ้นมาดังรูป ให้ทำการ Login ด้วย Classic Mode สำหรับรหัสผ่านที่ใช้นั้นกำหนดให้เป็น .2020kscc?


KSCC 2020 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Thu 22:57
kscc2020
Not listed?
CentOS
Right Control


KSCC 2020 [Running] - Oracle VM VirtualBox
File Machine View Input Devices Help
Tue 10:05
kscc2020
Password:
Cancel
Sign In
GNOME
GNOME Classic
Custom
CentOS
Right Control

การรันโปรแกรมที่เขียนขึ้น

MPI: ภายใน Virtual Machine ที่ได้รับไปนั้นจะมีตัวอย่างของ source code สำหรับการรันแบบ MPI อยู่ที่ /export/home/kscc2020/Desktop/SampleCode/ mpi_hello_world.c

สำหรับการรัน code ตัวอย่างให้ทำดังนี้

1. เปิดTerminal และเข้าไปที่ Directory SampleCode

```
[kscc2020@kscc2020 SampleCode]$ cd /export/home/kscc2020/Desktop/SampleCode/
[kscc2020@kscc2020 SampleCode]$ ls
mpi_hello_world  mpi_hello_world.c  openmp_hello  openmp_hello.c
[kscc2020@kscc2020 SampleCode]$
```

2. สำหรับคำสั่งที่ใช้สำหรับ compile mpi source code คือ “mpicc –o <output filename> <.c file> จากรูปด้านล่างคือ ต้องการ complile mpi_hello_world.c และต้องการ output เป็น file ชื่อว่า mpi_hello_word

```
[kscc2020@kscc2020 SampleCode]$ mpicc -o mpi_hello_world mpi_hello_world.c
[kscc2020@kscc2020 SampleCode]$
```

3. หลังจาก compile แล้วเราจะได้ binary file ชื่อว่า mpi_hello_world สำหรับการรัน จะใช้คำสั่งคือ mpirun –np <number of processor> <binary filename> ดังตัวอย่างคือ ต้องการรัน mpi_hello_world ด้วย 3 processor

```
[kscc2020@kscc2020 SampleCode]$ mpirun -np 3 mpi_hello_world
Hello world from processor kscc2020.innosoft, rank 0 out of 3 processors
Hello world from processor kscc2020.innosoft, rank 2 out of 3 processors
Hello world from processor kscc2020.innosoft, rank 1 out of 3 processors
[kscc2020@kscc2020 SampleCode]$
```

OpenMP: เช่นเดียวกับ MPI ภายใน Virtual Machine ที่ได้รับไปนั้นจะมีตัวอย่างของ source code สำหรับการรันแบบ OpenMP อยู่ที่ /export/home/kscc2020/Desktop/SampleCode/openmp_hello.c

สำหรับการรัน code ตัวอย่างให้ทำดังนี้

1. เปิดTerminal และเข้าไปที่ Directory SampleCode

```
[kscc2020@kscc2020 SampleCode]$ cd /export/home/kscc2020/Desktop/SampleCode/
[kscc2020@kscc2020 SampleCode]$ ls
mpi_hello_world  mpi_hello_world.c  openmp_hello  openmp_hello.c
[kscc2020@kscc2020 SampleCode]$
```

2. สำหรับคำสั่งที่ใช้สำหรับ compile mpi source code คือ “gcc –o <output filename> -fopenmp <.c file> จากรูปด้านล่างคือ ต้องการ compile openmp_hello.c และต้องการ output เป็น file ชื่อว่า openmp_hello

```
[kscc2020@kscc2020 SampleCode]$ gcc -o openmp_hello -fopenmp openmp_hello.c
```

หลังจาก compile แล้วเราจะได้ binary file ชื่อว่า openmp_hello สำหรับการรัน จะใช้คำสั่งคือ ./<binary filename> ดังตัวอย่างคือ ต้องการรัน openmp_hello สำหรับการระบุจำนวน thread ให้ใช้คำสั่ง “export OMP_NUM_THREADS=<number of thread>”

```
[kscc2020@kscc2020 SampleCode]$ ./openmp_hello
Hello World... from thread = 0
[kscc2020@kscc2020 SampleCode]$ export OMP_NUM_THREADS=5
[kscc2020@kscc2020 SampleCode]$ ./openmp_hello
Hello World... from thread = 2
Hello World... from thread = 0
Hello World... from thread = 1
Hello World... from thread = 4
Hello World... from thread = 3
[kscc2020@kscc2020 SampleCode]$
```