Math 4997-3 Quiz 8: Due by 2021/10/28

Exercises

1. Enhanced futures (4 credits):
   HPX implements following two additional `future` variants

   - `hpx::shared_future`
   - `hpx::make_ready_future`

   Please explain why HPX provides these additional future variants. Try to make a small example, where these are needed.

Programming exercise

- Using dataflow (4 credits):
  Replace the `hpx::when_all(futures).then()` by hpx::dataflow. Note that the function `square` is defined in the next source code listing.

```
run_hpx([](){

std::vector<hpx::lcos::future<int>> futures;
futures.push_back(hpx::async(square,10));
futures.push_back(hpx::async(square,100));

hpx::when_all(futures).then([](auto&& f){
    std::vector<hpx::lcos::future<int>> futures = f.get();
    int result = 0;
    for(size_t i = 0; i < futures.size();i++)
        result += futures[i].get();
    std::cout << result << std::endl;
});

});
```

- Unwrapping futures: (2 credits)
  In the code below are two comments and two lines of code are missing there. Please add the two missing lines of code.

```
// Evaluate the result and print the result
void sum(int first, int second){
        std::cout << first + second << std::endl;
}
```

```
// Compute the square
int square(int a)
{
return a*a ;
}

int main(){
        std::vector<hpx::lcos::future<int>> futures;
        futures.push_back(hpx::async(square,10));
        futures.push_back(hpx::async(square,100));

        // Unwrapp the function sum

        // Use dataflow to execute the function asynchronously.


        return 0;
}
```