운영체제 Assignmnet03

2021270660 이지원

System Description

```cpp
// 현재 시간을 출력하는 함수
void printCurTime() {
    SYSTEMTIME lt;
    // GetSystem() -> UTC 기준... 따라서 GetLocalTime()을 사용해 시간을 가져옴
    GetLocalTime(&lt); // get local time
    cout << lt.wYear << "-";
    cout.width(2);
    cout.fill('0');
    cout << lt.wMonth << "-";
    cout.width(2);
    cout << lt.wDay << "  ";
    cout.width(2);
    cout << lt.wHour << ":";
    cout.width(2);
    cout << lt.wMinute << ":";
    cout.width(2);
    cout << lt.wSecond << " -> ";
}
```

```cpp
// 현재 CPU 부하를 반환하는 함수
double cpu() {
    // 함수 사용을 위한 파라미터
    FILETIME idle, kernel, user;
    double r = 0;

    // t0에서의 CPU 사용 시간 구해오기
    GetSystemTimes(&idle, &kernel, &user);
    DWORD kernelHigh1 = kernel.dwHighDateTime;
    DWORD kernelLow1 = kernel.dwLowDateTime;
    DWORD userHigh1 = user.dwHighDateTime;
    DWORD userLow1 = user.dwLowDateTime;
    DWORD idleHigh1 = idle.dwHighDateTime;
    DWORD idleLow1 = idle.dwLowDateTime;


    Sleep(1000); // 1초 후

    // t1에서의 CPU 사용 시간 구해오기
    GetSystemTimes(&idle, &kernel, &user);
    DWORD kernelHigh2 = kernel.dwHighDateTime;
    DWORD kernelLow2 = kernel.dwLowDateTime;
    DWORD userHigh2 = user.dwHighDateTime;
    DWORD userLow2 = user.dwLowDateTime;
    DWORD idleHigh2 = idle.dwHighDateTime;
    DWORD idleLow2 = idle.dwLowDateTime;


    // 시간 계산
    // 실제 cpu 사용 시간 / 전체 시간 * 100
    double kernelTime = (kernelHigh2 + kernelLow2) - (kernelHigh1 + kernelLow1);
    double userTime = (userHigh2 + userLow2) - (userHigh1 + userLow1);
    double idleTime = (idleHigh2 + idleLow2) - (idleHigh1 + idleLow1);

    double delta = ((userTime + kernelTime) - idleTime) * 100 / (kernelTime + userTime);

    return delta;
}
```

System Description

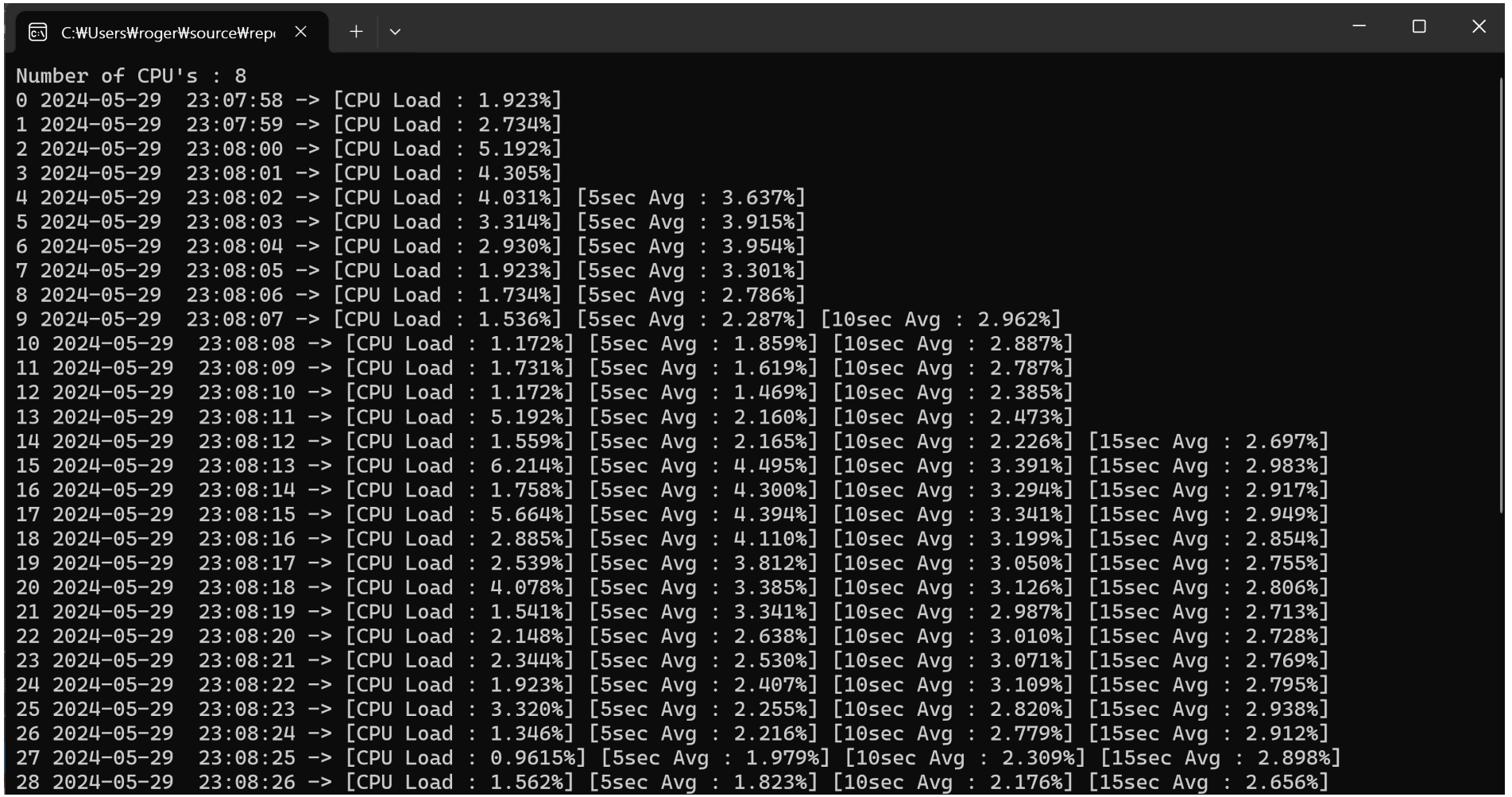
```
double getAvgCPU(double total[], int n, int k) {
   double  sum = 0;

   if (n == 15) {   // 처음부터 전체에 대한 평균일 경우
      for (int i = 0; i < n; i++) {
         sum += total[i];
      }
   } else if ( k < 14) { // 15개 미만의 데이터 존재, 최근 10개나 5개 ...
      for (int i = 0; i < n; i++) {
         sum += total[k - i];
      }
   } else { // 전체 중에 최근 10개, 5개만 평균
      for (int i = 0; i < n; i++) {
         int index = k % 15 - i;
         if (index < 0) {
            index = i;
         }
         sum += total[index];
      }
   }
   return sum / n;
}
```

```
void printNumCPU() {
   SYSTEM_INFO info;
   GetSystemInfo(&info);
   cout << "Number of CPU's : " << info.dwNumberOfProcessors << endl;
}
```

System Description

```
int main(void) {
   double total[15] = { 0, };
   int i = 0;
   int k = 0;

   // 프로세스 개수 출력
   printNumCPU();

   while (true) {
      if (i > 14) {
         k = i%15;
      }
      else {
         k = i;
      }

      total[k] = cpu();

      // 시간 출력
      cout << i << " ";
      printCurTime();

      // 현재 CPU Load 출력
      cout << "[CPU Load : " << showpoint << setprecision(4) << total[k] << "%]";

      // 평균 CPU Load 출력
      if ( i >= 4 ) {
         cout << " [5sec Avg : " << showpoint << setprecision(4) << getAvgCPU(total, 5, i) << "%]";
      }
      if ( i >= 9 ) {
         cout << " [10sec Avg : " << showpoint << setprecision(4) << getAvgCPU(total, 10, i) << "%]";
      }
      if ( i >= 14 ) {
         cout << " [15sec Avg : " << showpoint << setprecision(4) << getAvgCPU(total, 15, i) << "%]";
      }

      cout << endl;
      i++;
   }
}
```

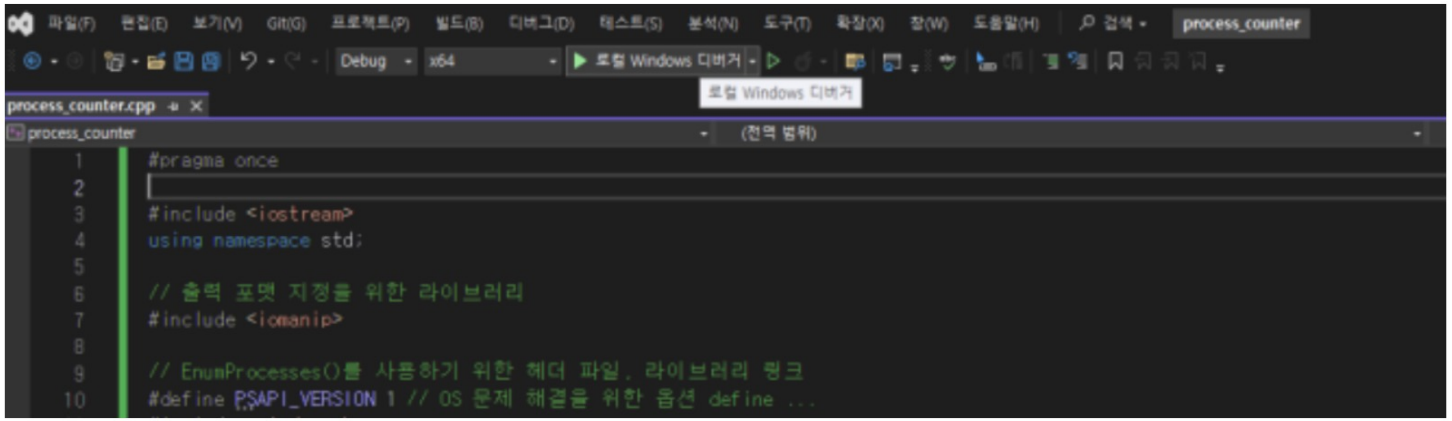
Test Description



User Documentation

1. .exe 파일 실행 (빌드된 상태)

2. 로컬 디버거로 실행 : 프로젝트 생성 후 소스 파일에 추가 후 로컬 디버거로 실행



3. 빌드 후 .exe 파일 실행 : 마찬가지로 프로젝트 생성 후 빌드 - 솔루션 빌드 후 .exe 파일을 찾아 실행