PC で開発したプログラムが大型計算機で make できない

★プログラムの概要：
PC 上で開発した数値計算プログラム

★助言、提案の欲しい問題：
local PC で開発し、正しく動作しているプログラムを大型計算機 (スーパーコンピュータ) 上で使おうとすると、make が通らなかったので、大型計算機上で make を通るようにし、PC で計算した場合と同じ結果が得られることを確認したい。

★使用する計算機（予定も含む）：
大型計算機 (BlueGene/Q)

★用いる言語：
C++

今回は、担当者が同じスーパーコンピュータのアカウントを持っていたので、ソースファイル一式を依頼者から提供していただき、問題点を調べることができました。その結果、以下のような点を修正することにより、無事 make を実行し、実行可能ファイルの生成を行うことができました。

[1] makefile の修正

大型計算機では、Operating System やインストールされているソフトウェアが PC とは異なる場合があります。そのため、PC 上で正常に動いていた makefile がそのままでは大型計算機で動かないということがあります。

(a) 大型計算機の利用説明書などの確認
大型計算機の場合、アカウントを持っていれば、「利用の手引き」や、「利用者講習会資料」などが提供されているので、それらを入手し、コンパイルコマンド、必要なオプションなどを確認し、makefile を修正します。

(b) 大型計算機で正常に動いている類似の設定ファイル (makefile) の利用
今回の場合は、BlueGene/Q 上での make がうまくいかないという問題でしたが、幸い、旧 BlueGene/L 上で類似のソースコードを正常にコンパイルできる makefile があったので、それを参考に、依頼者の makefile を書き換えることができました。

[2] 不要なソースファイルのコンパイルを抑制

今回のソースファイルは、複数のソースファイルが複数のディレクトリに分散して置かれており、その中の一部のソースファイルについて、最終的な実行可能ファイルの生成には不要であるにもかかわらずコンパイルを行おうとしてコンパイルエラーが起こっていました。そこで、makefile やソースファイルのファイル名 (拡張子) を変更して、不要なソースファイルが make の対象外となるようにしました。

[3] 改行コードの変換

一部のソースファイルの改行コードが、unix ではなく mac の形式になっており、そのソースファイルのコンパイルが正しく行われていないようでしたので、改行コードを unix 形式に変換しました。例えば、perl が使える場合には、以下のようなスクリプトファイル (conv.sh) を用いて、

```
$ ./conv.sh   file_mac.cpp > file_unix.cpp
```

で改行コードを変換することができます。

```
-----------------------------------
$ cat conv.sh
#!/bin/sh
# mac --> unix
perl -pe 's/\r/\n/g' ${1}
-----------------------------------
```

[4] 複素数の扱いを統一

依頼者のソースファイルは C++で書かれていましたが、複素数を C 言語の拡張版で記述している部分があるためにコンパイルエラーが起こっていたので、どちらかにそろえるようにしました。

BlueGene/Q では、以下の 2 種類の記述法のどちらかを選択すればよいようです。

- C++的な記述の例

```
-------------------------------------------------------------------------
#include <complex>

std::complex<double> z;           // 倍精度複素数型変数 z の宣言
std::complex<double> z(1.0,1.0); // z を宣言し (1.0+1.0i) で初期化
std::complex<double>(0.0,1.0);    // 倍精度複素数型定数の虚数単位 i
std::complex<double>(0.0,0.0);    // 倍精度複素数型定数のゼロ

real(z);                          // 実部
imag(z);                          // 虚部
conj(z);                          // 複素共役
-------------------------------------------------------------------------
```

- C99 的な記述の例

```
-------------------------------------------------------------------------
#include <complex.h>

double _Complex z;                // 倍精度複素数型変数 z の宣言
double _Complex z = (1.0+1.0i); // z を宣言し (1.0+1.0i) で初期化
(0.0 + 1.0i);                     // 倍精度複素数型定数の虚数単位 i
(0.0 + 0.0i);                     // 倍精度複素数型定数のゼロ

__real__(z);                      // 実部
__imag__(z);                      // 虚部
~(z);                             // 複素共役
-------------------------------------------------------------------------
```

以上 4 点についての修正を行った結果、無事 make コマンドを実行して実行可能ファイルの生成を行えるようになりました。