第 6 章 演習課題

課題 1

サンプルプログラムをコンパイル・実行して動作を確認せよ．さらに，適宜修正してその実行結果を確認せよ．

課題 2

以下のように掛け算九九の表を標準出力にキレイに表示するプログラムを作成せよ．

```
$ ./a.out
   1   2   3   4   5   6   7   8   9
   2   4   6   8  10  12  14  16  18
   3   6   9  12  15  18  21  24  27
   4   8  12  16  20  24  28  32  36
   5  10  15  20  25  30  35  40  45
   6  12  18  24  30  36  42  48  54
   7  14  21  28  35  42  49  56  63
   8  16  24  32  40  48  56  64  72
   9  18  27  36  45  54  63  72  81
```

課題 3

helix1.dat から実数データ $x_i, y_i, z_i (i = 1, \ldots, 32)$ を読み込み，全く同じフォーマットで標準出力に表示するプログラムを作成せよ．以下のようにリダイレクトでファイルに出力し，`diff` コマンドによってフォーマットが同じかどうかを確かめよ．

```
$ ./a.out > test.dat
$ diff helix1.dat test.dat
```

上記の `diff` コマンドを実行して，出力が何も無ければファイルが同一であることを意味する．

データファイルには i 行目に x_i, y_i, z_i の 3 つの実数データが記述されている． `open` を用いて読み込むこと．

課題 4

課題 3 と同様に読み込んだデータ x_i, y_i, z_i をそれぞれ別のファイル (例えば `x.dat`, `y.dat`, `z.dat`) にアスキー形式で出力せよ．各データの書式は helix1.dat のものと同一とする．

このとき以下のコマンドによって結果を確認できる．

```
$ paste -d" " x.dat y.dat z.dat > test.dat
$ diff helix1.dat test.dat
```

`helix1.dat` と `test.dat` が同一ファイルになっていれば (`diff` コマンドが何も出力しなければ) 良い．

課題 5

バイナリファイル helix2.dat を open を用いて開き，実数データ $x_i, y_i, z_i (i = 1, \ldots, 32)$ を読み込み，先ほどの helix1.dat と全く同じフォーマットで標準出力に表示するプログラムを作成せよ．

作成したプログラムを

```
$ ./a.out > test.dat
$ diff helix1.dat test.dat
```

のように実行し，結果が helix1.dat と同じになることを diff コマンドによって確かめよ．

なお，このバイナリファイルは form='unformatted' で open したファイルに (その装置番号を 10 として)

```
write(10) x
write(10) y
write(10) z
```

のように出力したものである． ただしここで， x, y, z はそれぞれ長さ 32 の倍精度の実数配列である．すなわち

```
real(8) :: x(32), y(32), z(32)
```

のように宣言されたものであると考えれば良い．

課題 6

Fortran のソースコードから，何らかの Fortran の命令文を含む行数 (コメントのみの行および空白行を除いた行数) を数えるプログラムを作成せよ．

入力はリダイレクトによって

```
$ ./a.out < chap06/sample5.f90
 Number of lines with valid fortran statement :           24
```

のようにすれば良い． (チェックが出来ればファイル名は何でもよい．)

なお，コメントのみの行は最初の空白以外の文字が ”！”である行，空白行は空白のみで表される行であるとして判定すれば良い．組込み関数 adjustl を用いると良い．

課題 7 †

Fortran の通常の unformatted バイナリファイルは一般には他の言語と互換性が無いが， ストリーム入出力を使うことで他の言語と同様にバイナリファイルを扱うことが出来る．ここでは C 言語で

```
// 配列サイズ
const int N = 10;

// 倍精度実数の配列
double x[N];
```

```
// x には 1.0 から 5.5 まで 0.5 刻みでデータを格納

// x に格納された倍精度実数を N 個分ファイルにバイナリで出力
fwrite(x, sizeof(double), N, fp);
```

のように生成した cbinary.dat を Fortran から読み込むプログラムを作成せよ．実行結果は例えば以下のようになる．

```
$ ./a.out
data read from binary.dat in stream access
1.00
1.50
2.00
2.50
3.00
3.50
4.00
4.50
5.00
5.50
```

なおこのデータを作るのに用いた C 言語のコードは mkbin.c である．

課題 8 †

Fortran の `unformatted` バイナリファイル helix2.dat をストリーム入出力を用いて読み込み，課題 5 と同様に出力するプログラムを作成せよ．ここで多くのコンパイラが `unformatted` の場合には実際のデータの前後に 4 バイトずつヘッダーとフッター（データのバイト数を表す整数）を付与するので，これらを読み飛ばす必要があることに注意せよ．

これを理解しておけば多言語からもデータの読み書きが可能である．例えば，C 言語では helix.c ， Python では helix.py が同じ動作をするプログラムになっている．(Python の場合は scipy がインストールされていれば `scipy.io.FortranFile` を使って簡単に読み込むことが出来る．)