物理学情報処理演習
5. C言語①

ver20150508

2015年5月8日

5.1 プログラム開発

5.2 C言語の構造

5.3 入出力

5.4 変数の型

参考文献

・やさしいC++ 第4版　高橋 麻奈 (著)
ソフトバンククリエイティブ

・プログラミング言語C++第4版
ビャーネ・ストラウストラップ, Bjarne Stroustrup, 柴田 望洋

・Numerical Recipes: The Art of Scientific Computing, Third Edition in C++

身内賢太朗

レポート提出：fsci-phys-jouhou@edu.kobe-u.ac.jp

5.1 プログラム開発

- プログラム開発の流れ（先週の作業を思い出しながら。）
  - 5.1.1 プログラム編集
    - ソースコードファイルを作る
  - 5.1.2 コンパイル
    - 文法エラーをチェック
  - 5.1.3 デバグ
    - ソースコードとコンパイルメッセージを照らし合わせて誤りを発見する。
  - 実行
    - 実行時エラーをチェック
    - 実行結果をチェック

演習4-4：C言語のコンパイルと実行

ブラウザで本日のページを開き、hello.cxx
ルを別名で保存」する。拡張子txtは
ウント名を選択する。

```
$cd ~/lesson4
```
lesson4に移動

```
$cp ../hello.cxx ./
```
コピー

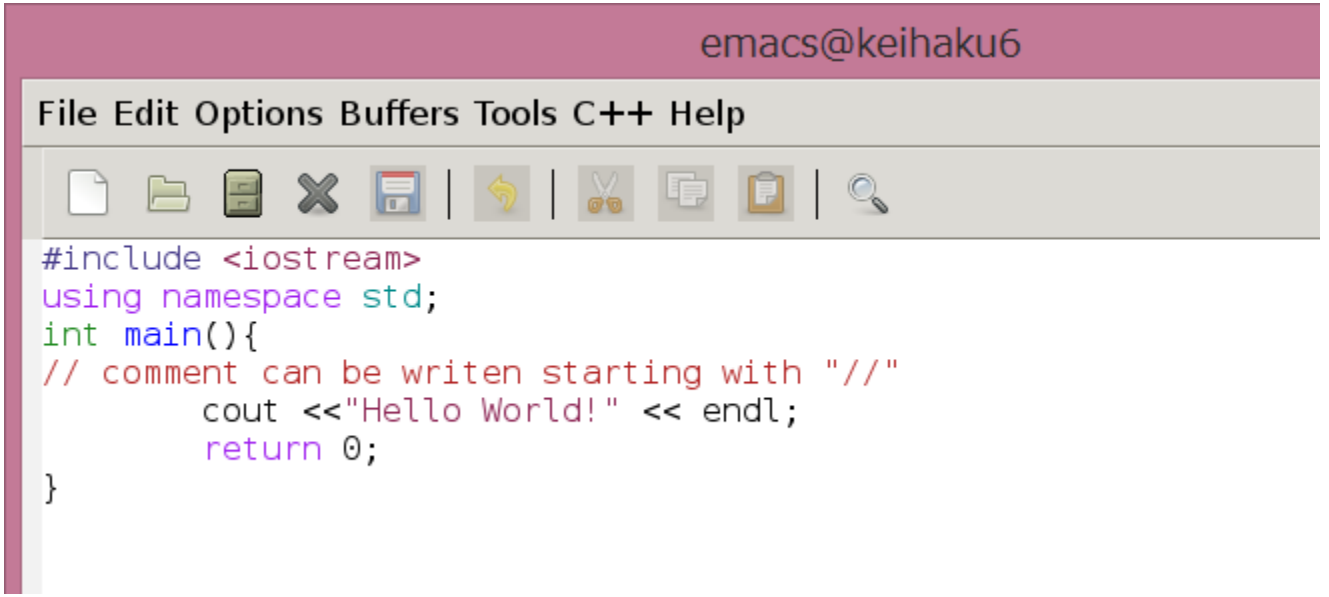
```
$ls
```
hello.cxxが存在することを確認

```
$g++ -o hello hello.cxx
```
コンパイル

```
$ls
```
helloが存在することを確認

```
$./hello
```

5.1.1 プログラム編集



- hello.cxx を眺める。

$mkdir lesson5

$cd lesson5

$cp ../lesson4/hello.cxx ./

$cat hello.cxx

　　表示

$emacs hello.cxx &

　　editorを用いて　hello.cxxを編集

　　editor:　テキストファイルの編集ソフト。純粋にテキストファイルをいじるための機能に特化している。書式情報などは保存、表示されない。

演習5.1.1 次ページ以降のコマンドを見ながらやってみよう(提出は不要)

emacs　の　ウィンドウ分割、戻す

別file名で保存　複数fileを開く　バッファー間の移動

行の先頭に移動　→　カーソルより右をカット　→　別の場所にペースト

文字列検索　　などなど。

エディターemacs コマンド集①

メニューバーから選ぶより圧倒的に編集効率が上がる。
この他にもコマンドあり。自分で調べて使い倒そう

☆☆☆　必須
☆☆　知ってると便利
☆　慣れたら覚えよう

- C-:「ctrl」を押しながらの意味
- M-:「esc」を押してから、もしくは「alt」を押しながらの意味

＜全般＞

☆☆☆　C-g　コマンドのキャンセル
☆☆☆　C-x C-c　Emacsの終了
☆☆　C-x u　最後の操作の取り消し
☆　C-z　Emacsの中断

＜ファイル関連＞

☆☆☆　C-x C-s 編集中のファイルのセーブ
☆☆　C-x C-w　別名のファイルにセーブ
☆☆　C-x C-f　ファイルを開く
☆☆　C-x i　別のファイルを挿入
☆☆　C-x k ファイルを閉じる
☆☆　C-x b　バッファー(開いているファイル)の移動
☆　C-x C-v　別のファイルに置き換える
☆　C-x s　編集中の全てのバッファをファイルの保存

＜ウィンドウ関連＞

☆☆　C-x 2　現在カーソルのあるウィンドウを上下に2分割
☆☆　C-x 1　現在カーソルのある他のウィンドウを1つ消す
☆☆　C-x o　現在カーソルのあるウィンドウ間のカーソルの移動
☆　C-x 0　現在カーソルのあるウィンドウを削除
☆　C-x 3　現在カーソルのあるウィンドウを垂直方向に分割

エディターemacs コマンド集②

- C-:「ctrl」を押しながらの意味
- M-:「esc」を押してから、もしくは「alt」を押しながらの意味

☆☆☆　必須
☆☆　知ってると便利
☆　慣れたら覚えよう

＜編集関連＞

☆☆☆　C-k　カーソルから行末までを消去
☆☆☆　C-y　最後に消去したものの再入
☆☆☆　C-s　文字列の検索
☆☆☆　C-r　文字列の検索(逆方向)
☆☆　C-d　カーソルの位置にある文字を削除
☆☆　M-k　カーソルから文末までを消去
☆☆　M-%　文字列の置換(yで置換, nでそのまま)
☆　C-space　範囲の開始位置を指定
☆　C-w　指定範囲のカット
☆　M-w　指定範囲のコピー
☆　M-x replace-string　文字列の一括
☆　C-x C-o　空行の一括削除

＜カーソル関連＞

☆☆☆　C-a　カーソルの位置を行頭に移動
☆☆☆　C-e　カーソルの位置を行末に移動
☆☆　M-<　ファイルの先頭に移動
☆☆　M->　ファイルの末尾に移動
☆☆　M-x goto-line　特定の行へ移動
☆　C-f　カーソルの位置を次の文字に移動
☆　C-b　カーソルの位置を前の文字に移動
☆　C-n　カーソルの位置を次の行に移動
☆　C-p　カーソルの位置を行頭に移動
☆　C-v　次の画面に移動
☆　M-v　前の画面に移動
☆　C-x C-x　マークした位置に移動. 再入力で元に戻る

＜日本語入力関連(入力方法依存あり)＞

☆☆　C-¥」: 日本語入力モード
C-n又はSPACE　次の変換候補を表示
C-p　前の変換候補を表示
C-i　変換対象を短くする
C-o　変換対象を長くする

5.1.2 コンパイル

- GNUのコンパイラ
  - gcc: Cコンパイラ
  - g++:C++コンパイラ

本演習ではC++を使用しますが、クラスを利用したオブジェクト指向というC++特有の機能までではカバーしません。C++でのみ許されている若干の文法表現のみ使用します。

i. pre-processor
ii. C-compiler
iii. assembler
iv. linker

全てを含んでいる

識別子により判断

.c: C言語ソース	i, ii, iii, iv
.cxx .C .cpp : C++言語ソース	i, ii, iii, iv
.h: ソース(ヘッダーファイル)	i, ii, iii, iv
.I: プリプロセス後のC言語ソース	i, ii, iii, iv
.o: オブジェクトファイル	iv

$ls
hello.cxxが存在することを確認

$g++ -o hello hello.cxx
コンパイル

$ls
helloが存在することを確認

$./hello

コピー

```
$ls
        hello.cxxが存在することを確認
$g++ -o hello hello.cxx
        コンパイル
$ls
        helloが存在することを確認
$./hello
```

- g++ のオプション
  - 書式　　gcc [option | filename]…

  よく使うオプション
  - -c:　　コンパイルまたはアセンブルまでで止める。コンパイラの出力はそれぞれのソースファイル(xxx.cxx)に対応したオブジェクトファイル(xxx.o)となる。
  - -o file:　出力先をfileに指定
    -o と fileの指定がないと　a.out　という実行fileができる。
  - -V:　　バージョン情報
  - -I dir:　dirをインクルードファイルの検索するディレクトリのリスト中に追加
  - -O1, -O2 …:　最適化を行う。数字が大きい方が最適化が深い

5.1.3 デバグ

- 文法間違いなどがあると、コンパイル時にメッセージ
  - error：重大な間違いでコンパイル未完了。要debug
  - warning：軽微な間違いなどでコンパイルは完了したが、想定通りに動作しない可能性あり。　→　debug推奨

- コンパイルは通るが、実行時にエラー

• コンパイル時のメッセージ例
（よくある順。必要に応じて確認のこと。）

〈Erorr〉

- parse error before ‘xxx’
  品詞解析エラー。括弧、セミコロン、クォーテーションのつけ
  間違いで、文の構造がおかしくなっているとき
- ‘xxx’ undeclared
  宣言せずに変数または型’xxx’が使われたとき
- redeclearation of ‘xxx’
  変数’xxx’を2回宣言したときinvalid value in assignment
  定数等、代入できないものに値を代入しようとしたとき。
- invalid operands to binary +
  ２項演算子 ‘+’ の使い方がおかしいとき
  例： x = x + “yy”;double とchar*は足せない
- array subscript is not an integer
  配列の添字が整数でないとき
- conflicting types for ‘xxx’
  関数’xxx’の宣言部と定義部で返り値の型、引数の個数・型が違うとき

- too many arguments to function ‘xxx’
- 関数’xxx’の引数の個数が多いとき
- incompatible type for argument i of “xxx”
- 関数’xxx’のi 番目の引数の型が違うとき
- undefined reference to ‘xxx’
- 関数’xxx’が定義されていないとき
- 定義がされていない、または関数名のミスタイプ
- ライブラリーがリンクされていない

〈Warning〉

- warning: assignment makes pointer from int without a cast
  整数をポインター型の変数に代入したとき
- warning: assignment from incompatible pointer type
  違う型へのポインター型の変数に変換したとき
- warning: control reaches end of non-void function
  返り値が’void’以外なのにreturn文が無いとき
- warning: unused variable ‘xxx’
  変数を宣言したのに使わなかったとき

• 実行時のメッセージ例
（よくある順。必要に応じて確認のこと。）

〈Erorr〉

•Segmentation fault (core dumped)
- 保護されているメモリー領域（コード領域、カーネル領域等）にアクセスを行った（SEGVとも書き表す）

•nan
- 負の数のsqrtを求めるなど、実行できない演算を行った。

•inf（または -inf）
- 0で割るなど無限大のでる演算を行った。

•Illegal instruction
- 不正な命令の発行

•Bus error
- ワード境界を超えてアクセスした。または外部バスのエラー

5.2 C言語の構造

- hello.cxx を眺める。
  $cat hello.cxx

```
#include <iostream>
using namespace std;
int main(){
// comment can be writen starting with "//"
        cout <<"Hello World!" << endl;
        return 0;
}
```

5.2 C言語の構造

- hello.cxx を眺める。
  $cat hello.cxx

5. 2. 1関数

hello.cxx

```
#include <iostream>
using namespace std;
int main(){
// comment can be writen starting with "//"
        cout <<"Hello World!" << endl;
        return 0;
}
```

- 関数
  - 文の集合で、特定の機能を持つ。
  - 型に応じた値を返す(return)。
  - { }で囲まれる。

  - プログラムは関数の集まりである。
  - 実行モジュールは必ずmain関数を持つ。

5.2.2 ライブラリ

hello.cxx

```
#include <iostream>
using namespace std;
int main(){
// comment can be writen starting with "//"
        cout <<"Hello World!" << endl;
        return 0;
}
```

- ライブラリ：特定の目的を持った関数の集合。
  #includeで読み込む
  - 〈iostream〉 入出力に関する関数
  - 〈stdlib.h〉 基本的なライブラリ
  - 〈string. h〉 文字列を扱う関数
  - 〈math. h〉 数学関係の関数

5.2.3 文

hello.cxx

```
#include <iostream>
using namespace std;
int main(){
// comment can be writen starting with "//"
        cout <<"Hello World!" << endl;
        return 0;
}
```

- 文
  - ;までが一文
  - 宣言文と実行文がある
    - 宣言文　変数の宣言
    - 実行分　　式、制御

5.2.4 書式

hello.cxx

```
#include <iostream>
using namespace std;
int main(){
// comment can be writen starting with "//"
    cout <<"Hello World!" << endl;
    return 0;
}
```

- インデント
  - 同じ階層の文は揃えて記述する。
  - TABで自動インデント可能

コメントは超重要。

- コメントアウト
  - 行の中で　//以降は　コンパイル時に無視される。
  - /**　と　**/　で囲む表記も可能。複数行にまたがるコメントも可能。

プログラムを書き換えるとき、
もともとの記述をコメントアウトしておくとすぐに元に戻せる。

5.3 入出力

- hello.cxx を眺める。
  $cat hello.cxx

```
#include <iostream>
using namespace std;
int main(){
// comment can be writen starting with "//"
        cout <<"Hello World!" << endl;
        return 0;
}
```

5.3 入出力

5.3.1 出力

hello.cxx

```
#include <iostream>
using namespace std;
int main(){
// comment can be writen starting with "//"
        cout <<"Hello World!" << endl;
        return 0;
}
```

- cout　標準出力
  - iostream の stdというnamespaceに定義されている。
  - 標準出力に出力するための関数。
  - コマンドラインでのリダイレクション　>　等でファイルに書き込める。
- cerr　標準エラー出力
  - 常に画面に表示される

参考file：hello_hello.cxx

演習5.3.1　（提出は不要）
hello.cxxの名前を変えてコンパイルしてみよう。　実行ファイルの名前も適当に変えること。
出力する文字を変えてみよう。　元の出力行はコメントアウトしておくこと。
coutとcerrを共存させ、ファイルへとリダイレクトしてみよう

5.3.2 入力

hello_cin.cxx

```
#include <iostream>
using namespace std;
int main(){
// comment can be writen starting with "//"
  char c[16];
  cout <<"Hello, input a message >";
  cin>>c;
  cout <<c << endl;
  return 0;
}
```

- cin　標準入力
  - 標準入力から入力するための関数。
  - ここではcという変数に入力している。

参考file：hello_cin.cxx

演習5.3.2　（提出は不要）
hello_cin.cxxをダウンロード、実行してみよう。

5.3.3 コマンドラインパラメータ

hello_com.cxx

```
#include <iostream>
using namespace std;
int main(int argc, char *argv[]){
// comment can be writen starting with "//"
  cout <<"Input a message with a command."<<endl;
  cout << "Number of command line parameters (argc)="<< argc << endl;
  cout << "Comand line paremeter 0 (argv[0]) : "<< argv[0] << endl;
  cout << "Comand line paremeter 1 (argv[1]) : "<< argv[1] << endl;
  return 0;
}
```

- main関数は引数を取ることができる。
  - 引数：実行時に コマンド名に続けて与えるパラメータ
  - argcにパラメータの数、argvには引数の内容が代入される。

参考file：hello_com.cxx

演習5.3.3　（提出は不要）
hello_com.cxxをダウンロード、実行してみよう。

実行時にパラメータを渡してみよう。

$./hello_com

$./hello_com hello

$./hello_com hello hellooo
の出力の違いを比べてみよう

5.4 変数

- 5.4.1　変数
  - 変数：”値”を入れておく箱
  - 変数は、
    - 名前 と 型 を持ち（変数定義）
    - 指し示した値を見たり（参照）
    - 値を変えたり（代入）

    できる。
  - 使用される変数は全て宣言する。

参考file：triangle_1.cxx

演習5.4.1
triangle_1.cxxをダウンロード、実行してみよう。

実行時にパラメータ50、プログラム中で20を渡す。

$./triangle_1 50

次ページからの説明を読みながら、このプログラムの内容を理解しよう。

triangle_1.cxx

```
#include <iostream>
#include <stdlib.h>
#include <string.h>
#include <math.h>
using namespace std;
int main(int argc, char *argv[]){
  //calculate the area of a triangle.
  //command line parameter is angle C(degree).
  int i=1;
  double a,b,c;
  double A,B,C,S;
  char message[128];

  a=10.0;

  strcpy(message,"input length b >");
  cerr <<"STEP"<< i <<": ";
  cout <<message;
  cin>>b;
  C=atof(argv[1]);
  cout << "a="<<a << " b="<<b<<" C="<<C<<"(degree)"<<endl;
  S=a*b*sin(M_PI*C/180)/2.;
  cout << "area="<<S<<endl;
  return 0;
}
```

triangle_1.cxx　：2辺とその間の角度から三角形の面積を求めるプログラム。

• 5.4.2　定数と変数

- 定数
  - 整定数:　　0xFF, 12
  - 浮動小数点整数:　　123.4, 1.5e-10
  - 文字定数:　　'a', '¥0', '¥xb'
  - 文字列:　　"Hello"
    - 文字定数と文字列は異なる
      - 'x' : 文字定数
      - "x" : 文字列(「x」と「¥0」から構成)　¥0:ヌル文字
- 定数と変数
  - 例

    int i, j;　　変数定義

    i = 3; 変数に定数を代入

    j = i * i;　　変数を参照、演算をし、結果を変数に代入

    9 = j; × 定数には代入できない

triangle_1.cxx

```
#include <iostream>
#include <stdlib.h>
#include <string.h>
#include <math.h>
using namespace std;
int main(int argc, char *argv[]){
  //calculate the area of a triangle.
  //command line parameter is angle C(degree).
  int i=1;
  double a,b,c;
  double A,B,C,S;
  char message[128];

  a=10.0;

  strcpy(message,"input length b >");
  cerr <<"STEP"<< i <<": ";
  cout <<message;
  cin>>b;
  C=atof(argv[1]);
  cout << "a="<<a << " b="<<b<<" C="<<C<<"(degree)"<<endl;
  S=a*b*sin(M_PI*C/180)/2.;
  cout << "area="<<S<<endl;
  return 0;
}
```

triangle_1.cxx　:2辺とその間の角度から三角形の面積を求めるプログラム。

• 5.4.2　変数の型

- 整数型
  - short　　16bit　　-2^{15}～ $2^{15}-1$　先頭bitは符号
  - int　　32 bit　　-2^{31}～ $2^{31}-1$　先頭bitは符号
  - long　　32(or 64) bit　先頭bitは符号
  - unsigned short　　16bit　　0～$2^{16}-1$
  - unsigned int　　32bit :　　0～$2^{32}-1$
  - unsigned long　　32(or 64)bit

- 浮動小数点型
  - float　　32bit
  - double　　64bit
  - long double 32bit

- 文字型
  - char　　8bit　1文字
  - 8bitのデータ　整数と見ることもできる。
  - signed char　　8bit　　-2^{7}～ $2^{8}-1$　先頭bitは符号
  - unsigned char　　8bit　　0～$2^{8}-1$

triangle_1.cxx

```
#include <iostream>
#include <stdlib.h>
#include <string.h>
#include <math.h>
using namespace std;
int main(int argc, char *argv[]){
  //calculate the area of a triangle.
  //command line parameter is angle C(degree).
  int i=1;
  double a,b,c;
  double A,B,C,S;
  char message[128];

  a=10.0;

  strcpy(message,"input length b >");
  cerr <<"STEP"<< i <<": ";
  cout <<message;
  cin>>b;
  C=atof(argv[1]);
  cout << "a="<<a << " b="<<b<<" C="<<C<<"(degree)"<<endl;
  S=a*b*sin(M_PI*C/180)/2.;
  cout << "area="<<S<<endl;
  return 0;
}
```

triangle_1.cxx　:2辺とその間の角度から三角形の面積を求めるプログラム。

• 5.4.2　変数への代入

triangle_1.cxx

- 定数
  - 定数とは、”値”
  - 定数には、以下のようなものがある
    - 整定数:　　0xFF, 12
    - 浮動小数点整数:　　123.4, 1.5e-10
    - 文字定数:　　‘a’, ‘¥0’, ‘¥xb’
    - 文字列:　　“Hello”
      - 文字定数と文字列は異なる
        - ‘x’：文字定数
        - “x”：文字列(「x」と「¥0」から構成)　¥0：ヌル文字
  - 定数と変数
    - 例

      int i, j;　　変数定義

      i = 3; 変数に定数を代入

      j = i * i;　　変数を参照、演算をし、結果を変数に代入

      9 = j; × 定数には代入できない

```
#include <iostream>
#include <stdlib.h>
#include <string.h>
#include <math.h>
using namespace std;
int main(int argc, char *argv[]){
  //calculate the area of a triangle.
  //command line parameter is angle C(degree).
  int i=1;
  double a,b,c;
  double A,B,C,S;
  char message[128];
  a=10.0;

  strcpy(message,"input length b >");
  cerr <<"STEP"<< i <<": ";
  cout <<message;
  cin>>b;
  C=atof(argv[1]);
  cout << "a="<<a << " b="<<b<<" C="<<C<<"(degree)"<<endl;
  S=a*b*sin(M_PI*C/180)/2.;
  cout << "area="<<S<<endl;
  return 0;
}
```

- 宣言時に代入することも、宣言と独立に代入することも可能。
- 一般には　=　を用いて代入するがstrcpyを用いた文字列への代入、cinによる標準入力からの代入なども可能。

5.5 演算

- 5.4.1　算術演算
  - +(和)　-(差)　*(積)　/(除)　%(整数同士の除算で余り)
  - *, /, %　が+,−よりも優先される。

- 5.4.2 数学演算
  - <math.h>をincludeする
  - sin(x)　　正弦 (radで与える)
  - cos(x)　　余弦(radで与える)
  - tan(x)　　正接(rad)
  - asin(x)　　arc sin
  - acos(x)　　arc cos
  - atan(x)　　arc　tan
  - exp(x)　　指数関数
  - sqrt(x)　　平方根
  - log(x)　　自然対数(底e)
  - log10(x)　　常用対数(底10)
  - pow(x,y)　　xのy乗
  - fabs(x)　　絶対値
  - M_PI　　円周率

triangle_1.cxx

```
#include <iostream>
#include <stdlib.h>
#include <string.h>
#include <math.h>
using namespace std;
int main(int argc, char *argv[]){
  //calculate the area of a triangle.
  //command line parameter is angle C(degree).
  int i=1;
  double a,b,c;
  double A,B,C,S;
  char message[128];

  a=10.0;

  strcpy(message,"input length b >");
  cerr <<"STEP"<< i <<": ";
  cout <<message;
  cin>>b;
  C=atof(argv[1]);
  cout << "a="<<a << " b="<<b<<" C="<<C<<"(degree)"<<endl;
  S=a*b*sin(M_PI*C/180)/2.;
  cout << "area="<<S<<endl;
  return 0;
}
```

triangle_1.cxx　:2辺とその間の角度から三角形の面積を求めるプログラム。

ソースコードファイル名：2015_jouhou_05_学籍番号の下4桁.cxx
出力結果ファイル名：2015_jouhou_05_学籍番号の下4桁.txt

課題5:C言語①

三角形の面積を計算するプログラムで以下の仕様を全て持つものを製作し、ソースコード及び出力結果を提出せよ。

①1辺とその両端の角度をコマンドライン引数として入力する。
②標準出力には、3辺の長さ、3角の大きさ、面積、円周率を出力する。
③1辺の長さは10　2角は30度と45度として実行を行い、出力を上記出力結果ファイルに記録せよ。

課題提出

- 宛先　fsci-phys-jouhou@edu.kobe-u.ac.jp
- 件名　2015-report05_**学籍番号**の下4桁
- 本文　学籍番号と名前
- 添付ファイル：
  - 2015_jouhou_05_学籍番号の下4桁.cxx
  - 2015_jouhou_05_学籍番号の下4桁.txt
- 締め切り　2014年5月22日（金）13：00