プロ演1::課題3::説明資料2

演習トップページ

→ 問3 → 説明資料2 → START ボタン

http://www.ced.is.utsunomiya-u.ac.jp

/lecture/2004/prog/p1/kadai3/lesson2/

１．補足及び復習

Visual C，Wordで空白を表示する方法

「デバッガー」ページのAPPENDIXを参照

最大値を計算するマクロ

```
#define MAX(x, y) ((x)<(y)?(y):(x))
```

フィボナッチ数を計算する再帰呼び出し関数

```
int f(int n) {
    if (n<3) return 1;        // n< 3
    return f(n-1) + f(n-2); // n>=3
}
```

２．前処理(プリプロセッサ)

コンパイルする前の処理

#include

ヘッダーファイルを取り込む

#define

定数やマクロを定義する

#if #else #ifdef ...

条件付きコンパイル等

3．前処理の応用例：assert

マクロの定義によって動作が変わる仕組み

開発版

```
// #define NDEBUG
#include <assert.h>
...
assert(式);
...
```

違いはここだけ

完成版

```
#define NDEBUG
#include <assert.h>
...
assert(式);
...
```

4．assertとデバッグ

実行可能ファイルにはコードがない！

NDEBUGが定義されていたら，何もしない！．

定義されていなかったら，式を評価する．

評価値が偽(0)の場合，行番号を出力し
プログラムを直ちに終了させる．
そうでない場合は，何もしない！．

コードは生成される

5. assertの使用例

フィボナッチ数を求める関数(開発版)

```
// #define NDEBUG
#include <assert.h>

int f(int n) {
  assert(n>=1); // n>=1がチェックする
  if (n<3) return 1;
  else return f(n-1) + f(n-2);
}
```

6. assertの練習(12分)

下記プログラムをテストしバッグを修正せよ

```
#include <stdio.h>
  フィボナッチ数を求める関数(前スライド参照)

int main(void) {
   int n;
   scanf(“%d”, &n);
   printf(“f(%d) = %d.\n”, n, f(n));
}
```

7．練習問題の正解

main()は入力をチェックしていなかった

完成版のmain関数

```
int main(void) {
  int n;
  scanf("%d", &n);
  if (n >= 1)
    printf(“f(%d) = %d.\n”, n, f(n));
  else
    printf(“f(%d)は無意味.\n”, n);
}
```

（黄色は追加分）

8．プログラム完成版

```
#include <stdio.h>
#define NDEBUG
#include <assert.h>

int f(int n) {
  assert(n>=1);
  return (n<3) ? 1 : (f(n-1)+f(n-2));
}

int main(void) {
  int n;
  scanf("%d", &n);
  if (n>=1)
    printf("f(%d) = %d.\n", n, f(n));
  else
    printf("f(%d)は無意味. \n", n);
}
```

実行可能ファイルでは
赤部分のないものと同じ

9. 配列

同じ型のデータの列で，a[i-1]のように
i番目のデータを使うことができる．例：

```
int a[10]; // 10個intの配列を定義
a[0] = 1;  // 一番目はa[0]
a[9] = 5;  // a[9]は10番目
scanf(“%d”, &a[1]); // a[1]を入力
printf(“%d”, a[2]); // a[2]を出力
```

１０．配列の使用例：総和計算

```
int a[10];   // 10個intの配列を定義
int sum = 0; // 総和，初期値は 0
for (i=0; i<10; i++) {
  scanf(“%d”, &a[i]); // a[i]の入力
  sum += a[i]; // つまりsum=sum+a〔i〕
}
printf(“総和 = %d.\n”, sum);
```

１１．配列使用時の注意点

int a[10] のようにサイズが定数である
(注：#define SIZE 10
　　int a[SIZE]; でも大丈夫)

C99は変数可

関数の引数が配列であるとき，普通
f(int n[])のように参照を渡す

詳細はポインタを勉強する時に

12. 学籍番号用check_digit()

```
char check_digit(int n[6]) {
  int weight[6] = {2, 3, 4, 5, 6, 7};
  int i, sum = 0;
  for (i=0; i<6; i++)
    sum += weight[i] * n[i];
  sum %= 11;
  ... // アルファベットへの変換
}
```

定義と共に初期化

１３．練習問題(15分)

ISBNコードの検証プログラムを作成せよ．
入力は，10桁のISBNコード(バラバラ可)．
正しいかどうかの判定結果を出力する．

重み	10	9	8	7	6	5	4	3	2	1
ISBN	4	3	2	0	0	2	6	9	2	6

← X は 10

OK

```
(10*4+9*3+...+2*2+1*6)%11 = 0
```

１４．配列の種類

intのほか，char，double，long，...
などの配列も使える

多次元配列

`int a[8][10];` のように定義すると，
`a[i][j] = k;` のように使える．

3次元以上もOK

１５．多次元配列の使用例

行列の和

```
#include <stdio.h>
int main(void) {
  int a[2][3], b[2][3], c[2][3], i, j;
  for (i=0; i<2; i++)
    for (j=0; j<3; j++) {
      scanf("%d%d", &a[i][j], &b[i][j]);
      c[i][j] = a[i][j] + b[i][j];
    }
  // cはaとbの和行列となる
}
```

１６．練習問題(19分)

行列の積を計算するプログラムを作成せよ．
ただし，サイズは三つとも#defineで定義する．
行列要素の入力は標準入力から行う．

ヒント：行列の積計算について
http://ja.wikipedia.org/wiki/行列