知能情報工学演習I
第12回（C言語第6回）
課題の回答

岩村雅一

masa@cs.osakafu-u.ac.jp

前回の課題1

- 球の体積を計算するマクロを作り、球の半径(小数とする)を入力したとき、球の体積を返すプログラムを作成しなさい。

```
#include<math.h>
#include<stdio.h>
#define V(r) 4.0/3.0*M_PI*r*r*r

int main(void){
  float i;
  printf("半径を小数で入力：");
  scanf("%f",&i);

  printf("半径%fの球の体積は%fです。
     ¥n",i,V(i));

  return(0);
}
```

マクロ中の
rをiに置き換える

課題１で実際にあった間違い（その１）

- 間違いではないが、ファイル名が日本語なのは文字化けして大変なのでやめてください
- 値を2回入力させられる

  scanf("%f¥n",&a);
- 答えが違う

  #define kyu(r) 4/3*(M_PI)*r*r*r

  #define seki(r) 3.0/4.0*M_PI*r*r*r;
- マクロを作っていない
  - 関数を作っている
  - main関数で計算

課題１で実際にあった間違い（その２）

- コンパイルが通らない（その１）

  printf("球の体積は%fです¥n",(4*M_PI*(r^3))/3);

- コンパイルが通らない（その２）
  - LaTeXの命令(¥begin{verbatim}など)が入っている（先週からずっと）
- 間違いではないが、M_PIを改めて定義している人がいる。/usr/include/math.hの中にM_PIが定義されているので、#include <math.h>とすればOK
  - #define M_PI 3.14159265358979323846

コメント

- 課題1のほうでgcc の後に-lm をつけなくてもコンパイルできたのですが、なぜでしょうか？
  - math.hに書いてあるもののうち、M_PIを使っているだけで、関数を使っていないから
- マクロが

  #define v(r) 4*M_PL*r*r*r/3

  だと

  error: ‘ M_PL ’ undeclared

と表示されてコンパイルできませんでした
  - M_PLではなく、M_PIです

前回の課題2

- 階乗(1からnまでの自然数の積)を計算する関数を作り、順列と組み合わせを表示しなさい
  - 順列

$$ {}_nP_r = n \ \ n-1 \ \ \ n-2 \ \cdots \ n-r+1 \ = \frac{n!}{n-r \ !} $$

  - 組み合わせ

$$ {}_nC_r = \frac{{}_nP_r}{{}_rP_r} = \frac{n!}{r! \ \ n-r \ !} $$

前回の課題の回答例1 (for文を使った場合)

```
#include <stdio.h>

/* 階乗を計算する関数 */
int fact(int x) {
  int i, fact = 1;
  for(i = 2; i <= x; i++) {
    fact *= i;
  }
  return(fact);
}
```

fact = 1 * 2 * 3 * 4 *…

0!や1!もok

```
int main (void){
  int n, r, p, c;

  printf("n: "); scanf("%d", &n);
  printf("r: "); scanf("%d", &r);

  p = fact(n) / fact(n-r);
  c = fact(n) / fact(n-r) / fact(r);

  printf("nPr = %d¥n", p);
  printf("nCr = %d¥n", c);

  return(0);
}
```

前回の課題の回答例2（関数の再帰的呼び出し）

main関数は説明のために消去

```
#include <stdio.h>

/* 階乗を計算する関数 */
int fact(int x) {
  if (x==1 || x==0) {
    return(1);
  } else {
    return(x*fact(x-1));
  }
}
```

```
int main (void){
  int n, r, p, c;

  printf("n: "); scanf("%d", &n);
  printf("r: "); scanf("%d", &r);

  p = fact(n) / fact(n-r);
  c = fact(n) / fact(n-r) / fact(r);

  printf("nPr = %d¥n", p);
  printf("nCr = %d¥n", c);

  return(0);
}
```

前回の課題の回答例2（関数の再帰的呼び出し）

例：fact(2)の場合

```
#include <stdio.h>

/* 階乗を計算する関数 */
int fact(int x) {
  if (x==1 || x==0) {
    return(1);
  } else {
    return(x*fact(x-1));
  }
}
```

fact(2)
=
fact(1)
=
1

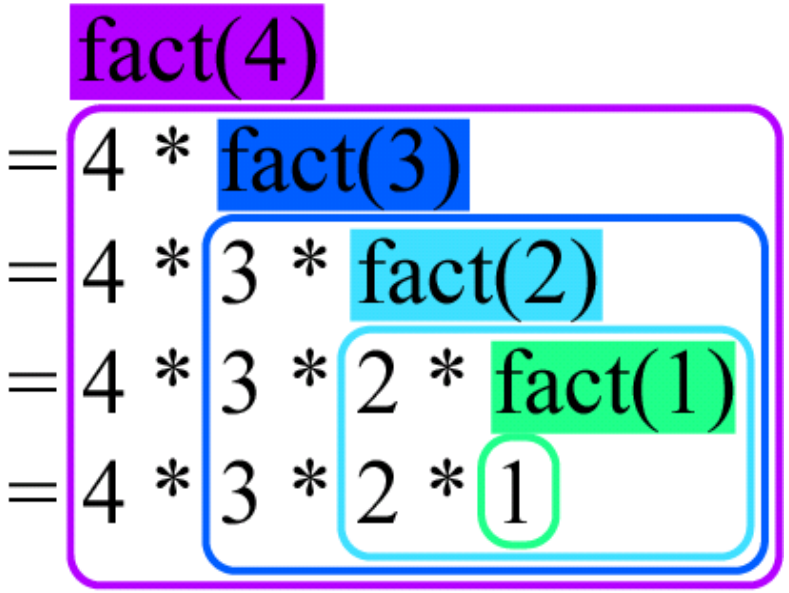
fact(1)の計算

```
/* 階乗を計算する関数 */
int fact(int 1) {
  if (x==1 || x==0) {
    return(1);
  } else {
    return(1*fact(1-1));
  }
}
```

前回の課題の回答例2
（関数の再帰的呼び出し）

```
#include <stdio.h>

/* 階乗を計算する関数 */
int fact(int x) {
  if (x==1 || x==0) {
    return(1);
  } else {
    return(x*fact(x-1));
  }
}
```



コメント・その１

- 課題二にかんして質問です。一からNまでの自然数の積を階乗とするなら、今回はゼロの階乗が一という定義は使いませんよね？
  - 使いませんが、関数の再利用性を考えると、0にも対応した関数を作成しておく方がいいでしょう。
- 文字化けに苦労しました。テフのソースの文字コードは何だったらいいんでしょう。

コメント・その２

- pdf が見付からなくて、かなり苦戦してしまいました……。
  - ごめんなさい、サーバートラブルに加えて、あの後すぐに出張だったのでアップロードが遅くなりました
- 課題の2 に関しては冗長になる気がして順列と組合せをまとめてしまったのですが、別々の入力で異なる変数を取った方が良かったのでしょうか。
  - 一緒の方が確かめるのが楽でいいです☺