# 前期中間試験問題　計算機応用　**5E**

**2004.06.17**

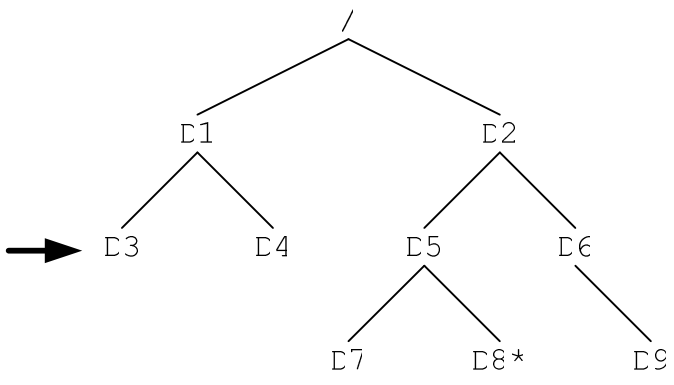
## 1．UNIX のコマンド

### 1.1 ファイル(各 1 点)

複数個のディレクトリー D1〜D9 が、図の構造で管理されている。各ディレクトリーには、複数のファイルやデータが存在する。*印のディレクトリ D8 (カレントディレクトリー) から矢印のディレクトリー D3 内のファイル f1 を指定したい。

(1) 絶対パスでの指定を書け。

(2) 相対パスでの指定を書け。




### 1.2 ディレクトリー(各 1 点)

(1) カレントディレクトリーを表す記号を示せ。

(2) 親ディレクトリーを表す記号を示せ。

(3) ディレクトリーの区切りを現す記号を示せ。

### 1.3 コマンド(各 1 点)

以下の場合のコマンドを書け。(各 1 点)

(1) カレントディレクトリーのパスを調べる。

(2) カレントディレクトリーにあるファイル名やサブディレクトリー名を調べる。

(3) 親ディレクトリーに移動する。

(4) サブディレクトリー hoge に移動する。

(5) ホームディレクトリーに移動する。

(6) 新たに hoge と言うサブディレクトリーを作成する。

(7) 空っぽのサブディレクトリー hoge を削除する。

(8) ファイルやサブディレクトリーがある hoge というサブディレクトリーを削除する。

(9) ファイル hoge を削除する。

(10) サブディレクトリー hoge を親ディレクトリーに huga と言う名前で移動する。

### 1.4 コンパイル・実行(1:3 点 2:各 2 点)

(1) C 言語のソースファイルを作成する場合、そのファイル名に重要な約束がある。その約束を記述せよ。

(2) C 言語のソースファイル test.c をコンパイルとリンクをして、実行ファイル testrun を作成したい。その場合のコマンドを書け。

[ア] test.c に数学関数がある場合

[イ] test.c に数学関数が無い場合

## 2．C 言語の文法

### 2.1 基礎(各 3 点)

(1) コメント文(注釈文)の書き方を示せ。

(2) コメント文を書く理由を簡潔に説明せよ。

### 2.2 変数(1:3 点 2:各 2 点 3:各 2 点)

(1) ローカル変数とグローバル変数の違いを説明せよ。

(2) 次のプログラムの中について、以下の問いに答えよ。

[ア] グローバル変数をすべて書き出せ。

[イ] ローカル変数をすべて書き出せ。

```
#include <stdio.h>
double func(double a);
double x;

/* ----------  main  ------------------*/
int main(){
   double y,theta;

    theta=3.1415924/6.0;
    y=func(theta);
    printf("%lf\t%lf\t%lf\n",theta,x,y);

   return(0);
}

/* ---------  func  ------------------*/
double func(double a){
   double b;

   b = sin(a);

    return(b);

}
```

(3) 以下の場合の変数宣言を示せ。1 行で完璧に記述すること。

［ア］ 文字型変数 aa と bb

［イ］ 整数型変数 i と j、k、l、m

［ウ］ 倍精度実数型変数 A と E、F

**2.3** 制御文(各 3 点)

(1) 次のように 2 つの場合に分けて、処理を行いたい。if 文を示せ。

・$100 \le i$ ならば

$$y = \sin(x)$$
$$i = 1$$

・$i < 100$ ならば

$$y = \cos(x)$$
$$i = -1$$

(2) for 文を用いて、1～100 まで足し合わせるプログラムを完成させよ。合計は、変数 sum に格納される。

```
#include <stdio.h>
int main(){
   int i,sum;

    sum=0;

   ここの文を書く

    printf("sum=%d\n",sum);

   return(0);
}
```

(2) while 文を用いて、1～100 まで足し合わせるプログラムを完成させよ。合計は、変数 sum に格納される。

```
#include <stdio.h>
int main(){
   int i,sum;

    sum=0;
    i=1;

   ここの文を書く

    printf("sum=%d\n",sum);

   return(0);
}
```

**2.4** 配列(1:3 点 2:各 2 点)

(1) 通常の変数に比べて、配列を使うと有利な場合を示せ。

(2) 以下の配列の宣言文を示せ。答えには、C 言語の完璧な 1 行を書くこと。配列名は、回答者が適当に決めよ。

［ア］ 整数型の 2 次元配列を用意する。ただし、配列の添え字は、0～100 と 0～100 とする。

［イ］ 倍精度実数型の 3 次元配列を用意する。ただし、配列の添え字は、0～10 と 0～50、0～100 とする。

**2.5** 関数(各 2 点)

(1) 次のプログラムに使われている関数名を示せ。

```
#include <stdio.h>
double f(double x, double y);
double g(double x, double y);

int main(){
   double x,y,u,v;

    x=2.0;
    y=2.0;

    u=f(x,y);
    v=g(x,y);

   return(0);
}

double f(double x, double y){
   return(x/(x*x+y*y));
}

double g(double x, double y){
   return(-y/(x*x+y*y));
}
```

(2) 先の問題で、実行される関数の順序を示せ。

**2.6** ファイル処理(各 2 点)

(1) ファイル(calresult)に、変数の値と三角関数の値を書き出している。ファイル処理に関係する下線［ア］～［オ］を示せ。(各 2 点)。

```
#include <stdio.h>
#include <math.h>
main(){
    [ア] *out

    double pi=4.0*atan(1.0);
    double theta, s, c, t;
    int i;

    out = [イ] (" [ウ] ","w");

    for(i=0;i<=100;i++){
     theta = i*pi/100;
     s = sin(theta);
     c = cos(theta);
     t = tan(theta);
      [エ] (out,"%f¥t%f¥t%f¥t%f¥n",theta,s,c,t);
    }

    [オ] (out);

}
```

## 3．プログラムの作成(各 15 点)

以下に示されたプログラムを C 言語で作成すること。

### 3.1 メッセージの表示

以下のメッセージを表示するプログラムを作成せよ。(5 点)

```
Hello world
Hello Akita
```

### 3.2 関数の利用(各 7 点)

関数を使う問題である。次の指示に従い、プログラムを作成せよ。(5 点)

・次の 2 つの関数を計算する。これらは、C 言語の関数を用いて、計算すること。

$$f(x) = x - \frac{x^3}{6} + \frac{x^5}{120} - \frac{x^7}{5040}$$

$$g(x) = 1 - \frac{x^2}{2} + \frac{x^4}{24} - \frac{x^6}{720}$$

・変数の値を $x$=0～π とし、その間を 100 等分して、関数の値を調べる。

・計算結果は、ディスプレイに、

x の値　　f(x)の値　　g(x)の値

と書き出す。

・プログラムに用いる関数名と変数名は、回答者が適当に決めてよい。

・これらの関数はどのような関数か?。得点は与えないが、気がついた人は、解答欄の端にでも記入してください。

### 3.3 ネピア数の計算とファイル出力(各 7 点)

ネピア数 $e$ の値は、マクローリン展開を用いると、以下の級数で表すことができる。

$$e = \sum_{i=0}^{\infty} \frac{1}{i!}$$

$$= \frac{1}{0!} + \frac{1}{1!} + \frac{1}{2!} + \frac{1}{3!} + \frac{1}{4!} + \frac{1}{5!} + \frac{1}{6!} + \frac{1}{7!} + \cdots$$

$$= 1 + 1 + \frac{1}{2\times1} + \frac{1}{3\times2\times1} + \frac{1}{4\times3\times2\times1} + \cdots$$

この式を用いて、ネピア数の値を計算するプログラムを作成する。ただし、条件は、以下の通りとする。

・右辺の計算する項数をひとつずつ増加させて、

$$e = 2.71828182845904523536028747135 2\cdots$$

に近づく事を調べる。

・右辺を 1 項まで、2 項まで、3 項まで・・・20 項までと項数を増やして計算する。計算した項数と値をディスプレイに書き出すとともに、ファイルへ書き込む。

・計算精度は、気にしないでプログラムを書け。