

2017年度 プログラミングII レポート10

学生用

学籍番号： _____

氏名： _____

下記の注意事項を守り、次ページ以降の問いに答え、レポートを完成させなさい。

提出期限： 2017年12月26日(火) 13:00まで

提出場所： 理学部棟 正面玄関内に設置のレポートボックス

注意事項：

- (1) このページを印刷し、必要事項を記入の上(学籍番号欄と氏名欄は2箇所あるので忘れずに記入すること)、レポートの表紙として提出すること。
- (2) コンピュータ端末室を利用する場合は、情報システム利用ガイドラインを厳守すること。**特に、コンピュータ端末室では飲食禁止である。**
- (3) クラスメイトのレポートを参考にしたり、クラスメイトと協力してレポートを作成した場合は、教員控の協力者氏名欄にクラスメイトの氏名を記入すること。これらの場合も、自分の言葉で表現し直すこと。**コピー禁止。**
- (4) プログラミングIIについて、あなたの声を聞かせてください(教員控の意見・質問欄に記入のこと)。気軽にどうぞ(成績には一切影響しません)。

出題者： 幸山 直人

出題日： 2017年12月13日(水)

----- 切り取り線 -----

2017年度 プログラミングII レポート10

教員控

学籍番号： _____

氏名： _____

協力者氏名： _____ , _____ , _____

レポート作成に要した時間： _____ . _____ 時間

意見・質問：

問1 下記ソースプログラム「ニュートン法」(report10_01.c)は、関数 $f(x)$ に対して方程式 $f(x) = 0$ の近似解をニュートン法を用いて求めるプログラムである (このプログラムでは $f(x) = x^2 - 2$ において $\sqrt{2}$ の近似値を求めている)。次の(1)~(4)の問いに答えなさい。ただし、初期値 (START) である x_0 には適当な値を与え、条件 $|x_n - x_{n-1}| < 0.0000001$ を満たしたとき十分な精度 (EPSIRON) の近似解が得られたものとする (小数点以下第7位までの近似解を求めなさい)。また、初期値によっては収束しない場合もあるので、変数 n の最大値 (MAX) を 30 とする。

● ニュートン法

report10_01.c

```
1: #include <stdio.h>
2: #include <stdlib.h>      /* for exit() */
3: #include <math.h>       /* for fabs() and pow() */
4:
5: #define EPSIRON 0.0000001
6: #define MAX 30
7: #define START 3.0
8:
9: double f(double x);
10: double df(double x);    ←関数を追記
11:
12: int main(void)
13: {
14:     double x = START, y;
15:     int n = 1;
16:
17:     y = x - f(x) / df(x);    ←追記
18:     while (fabs(x - y) >= EPSIRON) {
19:         n++;
20:         if (n > MAX) {
21:             printf("解が収束しませんでした ¥n");
22:             exit(EXIT_FAILURE);
23:         }
24:         x = y;                ←追記
25:         y = x - f(x) / df(x); ←追記
26:     }
27:     printf("Step %2d: %.7f¥n", n, y);
28:
29:     return 0;
30: }
31:
32: double f(double x)
33: {
34:     return pow(x, 2) - 2;
35: }
```

(1) ソースプログラム「ニュートン法」(report10_01.c)を完成しなさい。さらに、ソースプログラムをコンパイルし、実行可能ファイルが正しく動作するか確認しなさい。なお、作成したソースプログラムは印刷してレポートに添付すること。

ヒント：実行結果は「Step 6: 1.4142136」となる。

解答例 別紙を参照のこと。

(2) ソースプログラム「ニュートン法」(report10_01.c)の関数「f()」と関数「df()」を書き換えて、 $\sqrt[3]{2}$ の近似値を求めたい。 $\sqrt[3]{2}$ の近似値を求めるためのソースプログラム「ニュートン法」(report10_01a.c)を作成しなさい。さらに、ソースプログラムをコンパイルし、実行可能ファイルが正しく動作するか確認しなさい。なお、作成したソースプログラムおよび**実行結果**を印刷してレポートに添付すること。

解答例 別紙を参照のこと。

(3) 関数 $f(x) = x - \cos x$ に対して方程式 $f(x) = 0$ の近似解を求めたい。方程式 $f(x) = 0$ の近似解を求めるためのソースプログラム「ニュートン法」(report10_01b.c)を作成しなさい。さらに、ソースプログラムをコンパイルし、実行可能ファイルが正しく動作するか確認しなさい。なお、作成したソースプログラムおよび**実行結果**を印刷してレポートに添付すること。

ヒント：実数解は1つ。

解答例 別紙を参照のこと。

(4) 関数 $f(x) = 50x^5 - 150x^4 - 750x^3 + 1350x^2 + 2750x + 1$ に対して方程式 $f(x) = 0$ の近似解を全て求めたい。方程式 $f(x) = 0$ の近似解を求めるためのソースプログラム「ニュートン法」(report10_01c.c)を作成しなさい。なお、プログラムを実行した際、近似解を求めるために設定した初期値および実行結果から得られたステップ数と近似解を下表にそれぞれ記入しなさい。

ヒント：実数解は5つ(初期値をいろいろ変えて解を見つける)。

初期値	ステップ数	近似解
-4.0	7	-2.9860257
-1.0	5	-1.3965182
0.5	5	-0.0003637
2.5	4	3.0282326
4.0	7	4.3546750

解答例 初期値・ステップ数・近似解は、それぞれ上表の通りである。

問 2 サイコロ (整数 1~6 の乱数) を N 回転がし、各出目の回数を求めるソースプログラム「乱数 (サイコロ)」(report10_02.c) を作成しなさい (例えばサイコロを振る回数 N を「`#define N 10000`」とする)。さらに、ソースプログラムをコンパイルし、実行可能ファイルが正しく動作するか確認しなさい (各出目の回数がそれぞれおおよそ 6 分の 1 となり、擬似乱数が乱数としての正当性を持つことをチェックしなさい)。なお、作成したソースプログラムは印刷してレポートに添付すること。

注意 : 乱数 (`srand()`, `rand()`, `time()`) の使い方をマスターしましょう。

注意 : サイコロの出目をカウントするために配列 `s[6]` を準備する。

注意 : サイコロの出目は `switch` 文を使って振り分ける。

解答例 別紙を参照のこと。

問 3 モンテカルロ法を用いて円周率 π の近似値を求めるソースプログラム「モンテカルロ法 (円周率)」(report10_03.c) を作成しなさい (例えば点を打つ回数 N を「`#define N 1000000`」とする)。さらに、ソースプログラムをコンパイルし、実行可能ファイルが正しく動作するか確認しなさい。なお、作成したソースプログラムは印刷してレポートに添付すること。

解答例 別紙を参照のこと。

問 1 (1) の解答例 「ニュートン法」 (report10_01.c)

```
1: #include <stdio.h>
2: #include <stdlib.h>      /* for exit() */
3: #include <math.h>       /* for fabs() and pow() */
4:
5: #define EPSIRON 0.0000001
6: #define MAX 30
7: #define START 3.0
8:
9: double f(double x);
10: double df(double x);    ←関数を追記
11:
12: int main(void)
13: {
14:     double x = START, y;
15:     int n = 1;
16:
17:     y = x - f(x) / df(x);    ←追記
18:     while (fabs(x - y) >= EPSIRON) {
19:         n++;
20:         if (n > MAX) {
21:             printf("解が収束しませんでした ¥n");
22:             exit(EXIT_FAILURE);
23:         }
24:         x = y;                ←追記
25:         y = x - f(x) / df(x); ←追記
26:     }
27:     printf("Step %2d: %.7f¥n", n, y);
28:
29:     return 0;
30: }
31:
32: double f(double x)
33: {
34:     return pow(x, 2) - 2;
35: }
36:
37: double df(double x)        ←追記
38: {                            ←追記
39:     return 2 * x;          ←追記
40: }                            ←追記
```

問 1 (2) の解答例 「ニュートン法」 (report10_01a.c) * 変更部分のみ

```
⋮          ----- 続き -----  
  
31:  
32: double f(double x)  
33: {  
34:     return pow(x, 3) - 2;  
35: }  
36:  
37: double df(double x)  
38: {  
39:     return 3 * pow(x, 2);  
40: }
```

```
Z:¥src>report10_01a.exe   
Step 7:  1.2599210  
  
Z:¥src>
```

問 1 (3) の解答例 「ニュートン法」 (report10_01b.c) * 変更部分のみ

```
⋮          ----- 続き -----  
  
31:  
32: double f(double x)  
33: {  
34:     return x - cos(x);  
35: }  
36:  
37: double df(double x)  
38: {  
39:     return 1 + sin(x);  
40: }
```

```
Z:¥src>report10_01b.exe   
Step 6:  0.7390851  
  
Z:¥src>
```

問2の解答例 「乱数(サイコロ)」(report10_02.c)

```
1: #include <stdio.h>
2: #include <stdlib.h>      /* for rand() and srand() */
3: #include <time.h>       /* for time() */
4:
5: #define N 10000
6:
7: int main(void)
8: {
9:     int s[6] = {0, 0, 0, 0, 0, 0};
10:    int i;
11:
12:    srand((unsigned)time(NULL));
13:
14:    for (i = 0; i < N; i++) {
15:        switch((int)(rand() / (RAND_MAX + 1.0) * 6.0) + 1) {
16:            case 1:
17:                s[0]++;
18:                break;
19:            case 2:
20:                s[1]++;
21:                break;
22:            case 3:
23:                s[2]++;
24:                break;
25:            case 4:
26:                s[3]++;
27:                break;
28:            case 5:
29:                s[4]++;
30:                break;
31:            case 6:
32:                s[5]++;
33:                break;
34:        }
35:    }
36:    for (i = 0; i < 6; i++) {
37:        printf("%d:%d\n", i + 1, s[i]);
38:    }
39:
40:    return 0;
41: }
```

問3の解答例 「モンテカルロ法(円周率)」(report10_03.c)

```
1: #include <stdio.h>
2: #include <stdlib.h>      /* for rand() and srand() */
3: #include <time.h>       /* for time() */
4:
5: #define N 1000000
6:
7: int main(void)
8: {
9:     int i, n = 0;
10:    double x, y;
11:
12:    srand((unsigned)time(NULL));
13:
14:    for (i = 0; i < N; i++) {
15:        x = (double)rand() / RAND_MAX;
16:        y = (double)rand() / RAND_MAX;
17:        if (x * x + y * y <= 1) n++;
18:    }
19:    printf("π ≐ %.15f≐n", 4.0 * (double)n / N);
20:
21:    return 0;
22: }
```

参考：問1(4)のソースプログラム「ニュートン法」(report10_01c.c) * 変更部分のみ

```
⋮          ----- 続き -----
31:
32: double f(double x)
33: {
34:     return 50 * pow(x, 5) - 150 * pow(x, 4)
35:         - 750 * pow(x, 3) + 1350 * pow(x, 2) + 2750 * x + 1;
36: }
37:
38: double df(double x)
39: {
40:     return 250 * pow(x, 4) - 600 * pow(x, 3)
41:         - 2250 * pow(x, 2) + 2700 * x + 2750;
42: }
```