応用プログラミングＡ　第６回演習問題　クラスの詳細　解答例

問題 １　オブジェクトの関数への引き渡しと関数からの返し

```
#include <iostream>
#include <cmath>
using namespace std;

class Triangle {
    double a, b, c;
public:
    void set_a(double x) { a = x; }
    void set_b(double x) { b = x; }
    void set_c(double x) { c = x; }
    double get_a() { return a; }
    double get_b() { return b; }
    double get_c() { return c; }
    double area();
    void show();
};

double Triangle::area() {
    double s;

    s = (a+b+c) / 2;
    return sqrt(s*(s-a)*(s-b)*(s-c));
}

void Triangle::show() {
    cout << "辺a = " << a << "¥n";
    cout << "辺b = " << b << "¥n";
    cout << "辺c = " << c << "¥n";
    cout << "面積= " << area() << "¥n";
}

Triangle max_tri(Triangle ob1, Triangle ob2) {
    if (ob1.area() > ob2.area())
        return ob1;
    else
        return ob2;
}

int main() {
    Triangle obj1, obj2, obj3;

    obj1.set_a(3.0);
    obj1.set_b(4.0);
    obj1.set_c(5.0);
    obj2.set_a(5.0);
    obj2.set_b(12.0);
    obj2.set_c(13.0);

    cout << "三角形¥n";
    obj1.show();

    cout << "三角形¥n";
    obj2.show();

    obj3 = max_tri(obj1, obj2);
    cout << "大きい三角形¥n";
    obj3.show();

    return 0;
}
```

問題 ２　オブジェクトのアドレスの関数への引き渡し

```
#include <iostream>
#include <cmath>
using namespace std;

class Triangle {
    double a, b, c;
public:
    void set_a(double x) { a = x; }
    void set_b(double x) { b = x; }
    void set_c(double x) { c = x; }
    double get_a() { return a; }
    double get_b() { return b; }
    double get_c() { return c; }
    double area();
    void show();
};

double Triangle::area() {
    double s;

    s = (a+b+c) / 2;
    return sqrt(s*(s-a)*(s-b)*(s-c));
}

void Triangle::show() {
    cout << "辺a = " << a << "¥n";
    cout << "辺b = " << b << "¥n";
    cout << "辺c = " << c << "¥n";
    cout << "面積= " << area() << "¥n";
}

void mul_tri(Triangle *ob, double x) {
    ob->set_a(ob->get_a() * x);
    ob->set_b(ob->get_b() * x);
    ob->set_c(ob->get_c() * x);
}

int main() {
    Triangle obj;
    double a;

    obj.set_a(3.0);
    obj.set_b(4.0);
    obj.set_c(5.0);

    obj.show();

    cout << "何倍にしますか？:";
    cin >> a;
    mul_tri(&obj, a);

    obj.show();

    return 0;
}
```

問題３　関数とオブジェクト

```
#include <iostream>
#include <cmath>
using namespace std;

class Coord {
    double x;
    double y;
    double z;
public:
    void set_x(double u) { x = u; }
    void set_y(double v) { y = v; }
    void set_z(double w) { z = w; }
    double get_x() { return x; }
    double get_y() { return y; }
    double get_z() { return z; }
    void show();
};

void Coord::show() {
    cout << "(" << x << "," << y << "," << z << ")";
}

Coord mid_Coord(Coord ob1, Coord ob2);
double dist_Coord(Coord ob1, Coord ob2);

int main() {
    Coord p1, p2, p3;

    p1.set_x(1.0);
    p1.set_y(6.5);
    p1.set_z(-10.0);
    p2.set_x(-1.0);
    p2.set_y(1.5);
    p2.set_z(4.0);

    cout << "点A";
    p1.show();
    cout << "¥n";
    cout << "点B";
    p2.show();
    cout << "¥n";

    p3 = mid_Coord(p1, p2);

    cout << "ABの中点";
    p3.show();
    cout << "¥n";

    cout << "AB間の距離は" << dist_Coord(p1, p2) <<
"です¥n";

    return 0;
}

Coord mid_Coord(Coord ob1, Coord ob2) {
    Coord ob;

    ob.set_x((ob1.get_x()+ob2.get_x())/2);
    ob.set_y((ob1.get_y()+ob2.get_y())/2);
    ob.set_z((ob1.get_z()+ob2.get_z())/2);

    return ob;
}

double dist_Coord(Coord ob1, Coord ob2) {
    return sqrt((ob1.get_x()-ob2.get_x())
        * (ob1.get_x()-ob2.get_x())
        + (ob1.get_y()-ob2.get_y())
        * (ob1.get_y()-ob2.get_y())
        + (ob1.get_z()-ob2.get_z())
        * (ob1.get_z()-ob2.get_z()));
}
```

問題４　索敵ゲーム

```
// #define DEBUG_MODE
// 上記のコメントを外せばターゲットと敵の位置も表示さ
れる（デバッグ用）

#include <iostream>
#include <cstdlib>
#include <ctime>
using namespace std;

#define MAXW 4    // マップの幅
#define MAXH 4    // マップの高さ

class Unit {
    int x, y;
public:
    Unit(int a, int b);
    int get_x() { return x; }
    int get_y() { return y; }
    void move_l() { if (x > 0) x--; }   // 左に移動
    void move_r() { if (x < MAXW - 1) x++; }  // 右
に移動
    void move_u() { if (y > 0) y--; }   // 上に移動
    void move_d() { if (y < MAXH - 1) y++; }  // 下
に移動
};

Unit::Unit(int a, int b) {
    if (a >= 0 && a < MAXW && b >= 0 && b < MAXH) {
        x = a; y = b;
    }
    else {
        x = 0; y = 0;
    }
}

int dist_unit(Unit ob1, Unit ob2) {
    return abs(ob1.get_x() - ob2.get_x()) +
abs(ob1.get_y() - ob2.get_y());
}

int main() {
    int rx, ry, dst, drc;

    srand((unsigned int)time(NULL));

    // プレイヤーオブジェクト生成（初期位置決定）
    rx = rand() % MAXW;
    ry = rand() % MAXH;
    Unit ply(rx, ry);

    // ターゲットオブジェクト生成（初期位置決定）
    // プレイヤーユニットと同列・同行にならないように設定
    do {
        rx = rand() % MAXW;
        ry = rand() % MAXH;
    } while (rx == ply.get_x() || ry == ply.get_y());
    Unit trg(rx, ry);

    // エネミーオブジェクト生成（初期位置決定）
    do {
        rx = rand() % MAXW;
        ry = rand() % MAXH;
    } while (rx == ply.get_x() || ry == ply.get_y());
    Unit en1(rx, ry);
    do {
        rx = rand() % MAXW;
        ry = rand() % MAXH;
```

```
    } while (rx == ply.get_x() || ry == ply.get_y());
    Unit en2(rx, ry);

    while (1) {
        cout << "あなたは(" << ply.get_x() << "," <<
ply.get_y() << ")にいます¥n";
#ifdef DEBUG_MODE
        cout << "ターゲットは(" << trg.get_x() << ","
<< trg.get_y() << ")にいます¥n";
        cout << "敵1は(" << en1.get_x() << "," <<
en1.get_y() << ")にいます¥n";
        cout << "敵2は(" << en2.get_x() << "," <<
en2.get_y() << ")にいます¥n";
#endif
        dst = dist_unit(ply, trg);
        cout << "ターゲットは";
        if (dst <= 2)    // 2マス以内にいるか判定
            cout << "近くにいます¥n";
        else
            cout << "近くにいません¥n";
        dst = dist_unit(ply, en1);
        cout << "敵1は";
        if (dst <= 2)
            cout << "近くにいます¥n";
        else
            cout << "近くにいません¥n";
        dst = dist_unit(ply, en2);
        cout << "敵2は";
        if (dst <= 2)
            cout << "近くにいます¥n";
        else
            cout << "近くにいません¥n";

        do {
            cout << "どちらに移動しますか？（下-2、左
-4、右-6、上-8、移動しない-5）:";
            cin >> drc;
        } while (drc != 2 && drc != 4 && drc != 6 &&
drc != 8 && drc != 5);
        switch (drc) {
        case 2:
            ply.move_d(); break;
        case 4:
            ply.move_l(); break;
        case 6:
            ply.move_r(); break;
        case 8:
            ply.move_u(); break;
        }
        // ターゲットの移動
        drc = ((rand() % 5) + 1) * 2;
        switch (drc) {
        case 2:
            trg.move_d(); break;
        case 4:
            trg.move_l(); break;
        case 6:
            trg.move_r(); break;
        case 8:
            trg.move_u(); break;
        }
        // 敵の移動
        drc = ((rand() % 5) + 1) * 2;
        switch (drc) {
        case 2:
            en1.move_d(); break;
        case 4:
            en1.move_l(); break;
        case 6:
            en1.move_r(); break;
        case 8:
            en1.move_u(); break;
        }
        drc = ((rand() % 5) + 1) * 2;
        switch (drc) {
        case 2:
            en2.move_d(); break;
        case 4:
            en2.move_l(); break;
        case 6:
            en2.move_r(); break;
        case 8:
            en2.move_u(); break;
        }

        // 判定
        if (dist_unit(ply, trg) == 0) {
            cout << "おめでとう！¥nターゲットを捕まえ
ました¥n";
            break;
        }
        if (dist_unit(ply, en1) == 0 ||
dist_unit(ply, en2) == 0) {
            cout << "敵に捕まりました…¥nゲームオーバー
¥n";
            break;
        }
    }

    return 0;
}
```

別解答例

```
// #define DEBUG_MODE
// 上記のコメントを外せばターゲットと敵の位置も表示される（デバッグ用）

#include <iostream>
#include <cstdlib>
#include <ctime>
#include <cstring>
using namespace std;

#define MAXW 4    // マップの幅
#define MAXH 4    // マップの高さ

class Unit {
    int x, y;
public:
    Unit(int a, int b);
    int get_x() { return x; }
    int get_y() { return y; }
    void move_l() { if (x > 0) x--; }   // 左に移動
    void move_r() { if (x < MAXW - 1) x++; }  // 右に移動
    void move_u() { if (y > 0) y--; }   // 上に移動
    void move_d() { if (y < MAXH - 1) y++; }  // 下に移動
};

Unit::Unit(int a, int b) {
    if (a >= 0 && a < MAXW && b >= 0 && b < MAXH) {
        x = a; y = b;
    }
    else {
        x = 0; y = 0;
    }
}

int dist_unit(Unit ob1, Unit ob2) {
    return abs(ob1.get_x() - ob2.get_x()) + abs(ob1.get_y() - ob2.get_y());
}

int main() {
    int rx, ry, dst, drc;

    srand((unsigned int)time(NULL));

    // プレイヤーオブジェクト生成（初期位置決定）
    rx = rand() % MAXW;
    ry = rand() % MAXH;
    Unit ply(rx, ry);

    // ターゲットオブジェクト生成（初期位置決定）
    // プレイヤーユニットと同列・同行にならないように設定
    do {
        rx = rand() % MAXW;
        ry = rand() % MAXH;
    } while (rx == ply.get_x() || ry == ply.get_y());
    Unit trg(rx, ry);

    // エネミーオブジェクト生成（初期位置決定）
    do {
        rx = rand() % MAXW;
        ry = rand() % MAXH;
    } while (rx == ply.get_x() || ry == ply.get_y());
    Unit en1(rx, ry);
    do {
        rx = rand() % MAXW;
        ry = rand() % MAXH;
    } while (rx == ply.get_x() || ry == ply.get_y());
    Unit en2(rx, ry);

    int flag = 0;    // クリア表示判定のフラグ
#ifdef DEBUG_MODE
    flag = 1;
#endif
    while (1) {
        system("cls");
        if (dist_unit(ply, trg) == 0 || dist_unit(ply, en1) == 0 || dist_unit(ply, en2) == 0) {
            flag = 1;
        }
        for (int j = 0; j < MAXH; j++) {
            for (int i = 0; i < MAXW; i++)
                cout << "＋ー";
            cout << "＋¥n";
            for (int i = 0; i < MAXW; i++) {
                char ch[4];
                strcpy(ch, "  ");
                if (flag && trg.get_x() == i && trg.get_y() == j)
                    strcpy(ch, "Ｔ");
                if (ply.get_x() == i && ply.get_y() == j)
                    strcpy(ch, "Ｐ");
                if (flag && en1.get_x() == i && en1.get_y() == j)
                    strcpy(ch, "E1");
                if (flag && en2.get_x() == i && en2.get_y() == j)
                    strcpy(ch, "E2");
                cout << "｜";
                cout << ch;
            }
            cout << "｜¥n";
        }
        for (int i = 0; i < MAXW; i++)
            cout << "＋ー";
        cout << "＋¥n";

        // 判定
        if (dist_unit(ply, trg) == 0) {
            cout << "おめでとう！¥nターゲットを捕まえました¥n";
            break;
        }
        if (dist_unit(ply, en1) == 0 || dist_unit(ply, en2) == 0) {
            cout << "敵に捕まりました…¥nゲームオーバー¥n";
            break;
        }

        dst = dist_unit(ply, trg);
        cout << "ターゲットは";
        if (dst <= 2)    // 2マス以内にいるか判定
            cout << "近くにいます¥n";
        else
            cout << "近くにいません¥n";
        dst = dist_unit(ply, en1);
        cout << "敵1は";
        if (dst <= 2)
            cout << "近くにいます¥n";
        else
            cout << "近くにいません¥n";
        dst = dist_unit(ply, en2);
        cout << "敵2は";
        if (dst <= 2)
            cout << "近くにいます¥n";
```

```
        else
            cout << "近くにいません¥n";

        cout << "    ８¥n";
        cout << "    ↑¥n";
        cout << "４←５→６¥n";
        cout << "    ↓¥n";
        cout << "    ２¥n";
        do {
            cout << "どちらに移動しますか？ ";
            cin >> drc;
        } while (drc != 2 && drc != 4 && drc != 6 &&
drc != 8 && drc != 5);
        switch (drc) {
        case 2:
            ply.move_d(); break;
        case 4:
            ply.move_l(); break;
        case 6:
            ply.move_r(); break;
        case 8:
            ply.move_u(); break;
        }
        // ターゲットの移動
        drc = ((rand() % 5) + 1) * 2;
        switch (drc) {
        case 2:
            trg.move_d(); break;
        case 4:
            trg.move_l(); break;
        case 6:
            trg.move_r(); break;
        case 8:
            trg.move_u(); break;
        }
        // 敵の移動
        drc = ((rand() % 5) + 1) * 2;
        switch (drc) {
        case 2:
            en1.move_d(); break;
        case 4:
            en1.move_l(); break;
        case 6:

            en1.move_r(); break;
        case 8:
            en1.move_u(); break;
        }
        drc = ((rand() % 5) + 1) * 2;
        switch (drc) {
        case 2:
            en2.move_d(); break;
        case 4:
            en2.move_l(); break;
        case 6:
            en2.move_r(); break;
        case 8:
            en2.move_u(); break;
        }
    }

    return 0;
}
```