

6.2. スレッド対応（月面着陸）

月面着陸シミュレーションプログラムをスレッド対応で作成します。
以下のサイトの問題から実装します。

<https://kitako.tokyo/lib/CTask.aspx?id=6>

北ソフト工房 各種ソフトウェア開発／プログラミング教育／コンサルタント

Top > C言語、Java 練習プログラム集

6. 月面軟着陸ゲーム

古くからあるコンピューター・ゲームです。

月着陸船を、無事、月面に軟着陸させてください。

月の重力は地球の約 1/6 なので、落下速度が毎秒 1.62 m/s 増加します。1 秒毎にエンジンを燃焼させて落下速度を調整します。燃料を 1 ユニット燃焼すると 0.1 m/s 落下速度が減少するものとします。1 回で最大 50 ユニットまで燃焼できます。

高度 100 m から降下を始め、月面に 1.0 m/s 以下の速度で到達すると着陸成功です。

- 高度方向を横軸にして（90 度回転させて）着陸船の位置を表示します。
- 燃焼ユニット数を入力したら、以下のように落下速度と高度を計算します。
 - 落下速度 += 1.62 // 月の重力加速度
 - 落下速度 -= (0.1 × 燃焼ユニット数) // エンジン燃焼
 - 高度 -= 落下速度 // 高度の計算

このプログラムが実体化されると、8ビットの数字列の回転表示を行います。別スレッドからインタラプトをかけることで回転を止め、その時点での月着陸船の状態を計算して表示したのち、さらに回転を続けます。条件を多少緩和するために着陸条件を以下のように変えます。

【着陸条件】

1. 高度 0 メートル以上4m未満
2. 落下速度 0 m/s以上10m/s以下
3. 初期燃料1000ユニット

(初期燃料は起動クラスから指定、また残燃料の確認も起動クラスで実施)

【実行例】

```
ca. MoonBaseThreadMain - C:\Users\mitsu\AppData\Local\Temp\run1.bat
[return]又は5秒後に再開始します
燃烧ユニット=入力値×10unit
何ユニット燃烧しますか(1~9で降下率増減 0で上昇) ?
11000100
高度 :13.680m 落下速度:5.920m/s 残燃料 :800unit
>■
[return]又は5秒後に再開始します
燃烧ユニット=入力値×10unit
何ユニット燃烧しますか(1~9で降下率増減 0で上昇) ? 2
00111111
高度 :8.140m 落下速度:5.540m/s 残燃料 :780unit
>■
[return]又は5秒後に再開始します
燃烧ユニット=入力値×10unit
何ユニット燃烧しますか(1~9で降下率増減 0で上昇) ?
10010010
高度 :2.980m 落下速度:5.160m/s 残燃料 :760unit
>■
OK!!! 着陸しました
-- Press any key to exit (Input "c" to continue) --
```

【ソースコード】 MoonBaseThread.java

```
public class MoonBaseThread extends Thread{  
    double fallSpeed,altitude;↓  
    int fuelUnit,burningUnit;↓  
  
    public MoonBaseThread(){↓  
        fallSpeed=0.0;↓  
        fuelUnit=250;↓  
        burningUnit=0;↓  
        altitude=100;↓  
    }↓  
  
    public void run(){↓  
        while(true){↓  
            try{↓  
                rolling();↓  
                Thread.sleep(70);↓  
            } catch (InterruptedException ie) {↓  
                calc();↓  
                result();↓  
                if( (altitude < 4 && altitude >= 0) && (fallSpeed <= 10 && fallSpeed >= 0) ){↓  
                    position();↓  
                    System.out.println("¥nOK!!! 着陸しました");↓  
                    break;↓  
                    // System.exit(0);↓  
                }↓  
                if( altitude < 0 ){↓  
                    // System.out.println("¥nNG... 激突しました");↓  
                    break;↓  
                    System.exit(0);↓  
                }↓  
                try {↓  
                    position();↓  
                    ↓  
                    System.out.print("¥n[return]又は5秒後に再開始します¥n");↓  
                    System.out.print("燃烧ユニット=入力値×10unit¥n");↓  
                    System.out.print("何ユニット燃焼しますか(1~9で降下率増減□0で上昇)? ");↓  
                    Thread.sleep(5000);↓  
                    System.out.println("¥n");↓  
                }catch(InterruptedException e){↓  
                    // e.printStackTrace();↓  
                    // System.out.println("リターンキー連打でここに入る");↓  
                }↓  
            }↓  
        }↓  
    }↓  
  
    public void rolling(){↓  
        int numA=new java.util.Random().nextInt(2);↓  
        int numB=new java.util.Random().nextInt(2);↓  
        int numC=new java.util.Random().nextInt(2);↓  
        int numD=new java.util.Random().nextInt(2);↓  
        int numE=new java.util.Random().nextInt(2);↓  
        int numF=new java.util.Random().nextInt(2);↓  
        int numG=new java.util.Random().nextInt(2);↓  
        int numH=new java.util.Random().nextInt(2);↓  
  
        ↓  
        System.out.printf("%b¥b¥b¥b¥b¥b¥b¥b¥d¥d¥d¥d¥d¥d¥d¥d",numA,numB,numC,numD,numE,  
    }↓  
  
    public void result(){↓  
        System.out.printf("高度 :%.3fm□",altitude);↓  
68      System.out.printf("落下速度:%.3fms□",fallSpeed);↓  
69      System.out.printf("残燃料 :%dunit",fuelUnit);↓  
70      System.out.println("");↓  
71  
72 }
```

```

63 public void position(){↓
64     for(int i=1;i<=(int)altitude/2;i++){↓
65         System.out.print(" ");↓
66     }↓
67     System.out.println(">■");↓
68 }↓
69 public void calc(){↓
70     fallSpeed += 1.62;↓
71     fallSpeed -= (0.1*burningUnit);↓
72     altitude -= fallSpeed;↓
73     fuelUnit -= burningUnit;↓
74 }↓
75 public double getAltitude() {↓
76     return altitude;↓
77 }↓
78 public double getFallSpeed() {↓
79     return fallSpeed;↓
80 }↓
81 public int getFuelUnit() {↓
82     return fuelUnit;↓
83 }↓
84 public void setBurningUnit(int b) {↓
85     this.burningUnit=b;↓
86 }↓
87 public void setFuelUnit(int f) {↓
88     this.fuelUnit=f;↓
89 }↓
90 }↓

```

【ソースコード】 MoonBaseThreadMain.java

```

1 public class MoonBaseThreadMain{↓
2     public static void main(String[] args){↓
3         MoonBaseThread mbt = new MoonBaseThread();↓
4         mbt.setFuelUnit(1000); //燃料の量はメインメソッドで設定↓
5         mbt.start();↓
6         while(true){↓
7             String s=new java.util.Scanner(System.in).nextLine();↓
8             if (s.equals("0")){mbt.setBurningUnit(0);}↓
9             if (s.equals("1")){mbt.setBurningUnit(10);}↓
10            if (s.equals("2")){mbt.setBurningUnit(20);}↓
11            if (s.equals("3")){mbt.setBurningUnit(30);}↓
12            if (s.equals("4")){mbt.setBurningUnit(40);}↓
13            if (s.equals("5")){mbt.setBurningUnit(50);}↓
14            if (s.equals("6")){mbt.setBurningUnit(60);}↓
15            if (s.equals("7")){mbt.setBurningUnit(70);}↓
16            if (s.equals("8")){mbt.setBurningUnit(80);}↓
17            if (s.equals("9")){mbt.setBurningUnit(90);}↓
18            if (s.equals("")){↓
19                System.out.println("");↓
20                mbt.interrupt();↓
21                if(mbt.getFuelUnit()<=0){↓
22                    System.out.println("<<<燃料がありません>>>");↓
23                    break;↓
24                }↓
25            }↓
26        }↓
27        mbt.stop();↓
28    }↓
29 }↓

```

