2008 年

問題 1

(a)

```
def reverse(x,p,q)
    for i in p..(p+q)/2
        a=x[i]
        x[i]=x[p+q-i]
        x[p+q-i]=a
    end
    return x
end
```

(b)

```
def transpose1(x,k)
    for i in 1..k
        rotate(x,0,x.length-1)
    end
    return x
end
```

一回の rotate にかかる計算量は O(n)であるため、
総計算量は、k×O(n)=O(kn)である。(k は定数ではないから。)

(c)

```
def transpose2(x,k)
    reverse(x,0,k-1)
    reverse(x,k,x.length-1)
    reverse(x,0,x.length-1)
end
```

一回の reverse にかかる計算量は O(n)であるため、
総計算量は、3×O(n)=O(n)である。

解説

reverse を定義するには、以前レポートでよく使った、swap 関数に用いられる手法を使うと書きやすいかと思います。transpose1 は rotate を k 回繰り返しているだけ、transpose2 は reversse を 3 回使うというヒントがあるため、特に困ることはないでしょう。計算量もとりたてて難しいところはありません。

問題 2

(a) 出現確率が値によらず一定になっている数。

(b) 乱数のように見えるが前の値と無関係とは言えない、計算によって確率的に求めることのできる数。

(c) 厳密な解が存在し、それを乱数によって近似的に求める決定論的問題や、問題の中に確率的変動を含み、実験が困難であったり多大な費用がかかったりするために、乱数を用いて擬似的な実験を行う非決定論的問題に用いられる。

(d) 均一には分布しない。このプログラムでは円の半径が乱数になっているが、円の面積は半径の 2 乗に比例するため、このままでは内側ほどたくさん点が集まってしまう。よって、正しいプログラムに直すためには、半径の 2 乗を乱数にする、すなわち、r=sqrt(rand())とすればよい。

解説

授業ではやったけれど、レポートでは出なかったモンテカルロ法の問題です。記述はただ覚えておくだけ、間違いの指摘も面積を考えるという点に到れば、さほど難しいことはないはずです。

問題 3

(a) 11 回

(b) (a)の例で rec(a,5)を計算する時に、i=1 で rec(a,2)を呼び出しているが、この値を記録しておくことができないため、rec(a,4)を計算する時に i=0 で呼び出された rec(a,2)も再び計算することになってしまう。つまり、関数が呼び出された際にそれ以前の関数呼び出しの情報を利用することができないために、重複が起こる。

(c)(ア) 0　(イ)k　(ウ)n

(d) b[m]は、a の配列で与えられた要素で m を作る場合の、要素を使う回数の総和の最小値を表しているが、その値が配列に記録されているため、次に今までの最短回数を呼び出す際に、配列に記録しておいた b[m]を呼び出すことができるから。

解説

この問題はぶっちゃけかなり難しいです。このプログラムが何をしようとしているかというと、n を配列の要素で作ることのできる、全ての手数について調べているわけです。
最初の方のプログラムは、if　r>=0 and (k<0 or r+1<k) という部分が全てです。r>=0 によって、値が-1 であるもの、すなわち配列の要素で作ることのできないものをはじいています。全ての手数を調べるといっても、意味のないもの除去するわけです。k<0 というのは、for 文一回目なら・・・という意味です。r>=0 の条件が満たされた時点で、候補になる可能性が残されているため、とりあえず計算を進めるわけです。r+1<k というのは、既に入っている k という値と、新しく k に入れようとする値を比べているということです。最小値を求めるのですから、小さい方の値が採用されます。

次のプログラムも結局はほとんど同じことなのですが、m=1..n まで順番に m を配列の要素で作る場合の最短の手数を求め、それを b という配列に記憶させているという違いがあります。そのため、最初のプログラムと違い b[m-a[i]]がでてきたときにいちいち最初まで戻る必要がなく、配列に記録させた値をとりだしてくることによって、重複を避けることができるという利点があります。

この問題がさくさく解けたら怖いものなしです！

問題 4

(a) 配列の順序を入れ替える新たな関数を定義し、得点の配列を整列するプログラム による配列の入れ替えが行われるのと同時に、氏名と学生証番号の配列の順序を入れ替えればよい。

(b) 氏名データ(文字列)、学生証番号データ(整数)、得点データ(整数)

(c)・コンストラクタ
クラスからオブジェクトを生成すると自動的に実行され、インスタンス変数の初期化が行われる。

・アクセサ
インスタンス変数を読み書きできるようにする。

・得点整列インスタンスメソッド
得点データ順に配列の要素を並び替える。

(d)オブジェクト内の得点データインスタンス変数のみに注目して整列を行えばよい。

(e) (a)では、得点と同時に整列するために、配列の順序を入れ替える新たな関数を定義しなくてはならなかったが、(b)では、オブジェクトとしてまとまっているため、1 つのインスタンス変数を整列するとそれに伴い他のインスタンス変数も整列されるため、新たな関数を定義する必要がないという点で(b)の方が優れている。

解説

オブジェクト指向に関する記述問題です。インスタンス変数 3 つがセットになったオブジェクトが生成されるため、1 つのインスタンス変数についてソートすれば、オブジェクト全体が並び替えられるため、他の二つのインスタンス変数の順序も同時に並び替えられるという点がポイントです。授業スライドを見直して、オブジェクトに関する理解を深めておくとよいかと思います。