「最小交換貨幣枚数」による上手な払い方

情報処理学会・学会誌「情報処理」
2024年2月21日 13:46

冨澤眞樹（前橋工科大学）

令和7年度大学入学共通テスト試作問題「情報I」の第3問を取り上げる．第3問は，釣り銭に関する問題であり，アルゴリズムに関連した問題である．キーワード “お釣り 最適化” や “お釣り 最小” でネット検索すれば，釣り銭に関する問題やアルゴリズムに関するウェブサイトが多く見つかる．釣り銭問題は馴染みのある問題なのだろう．しかし，電子マネーやバーコード決済を使っている受験生に

は，釣り銭問題は馴染みが薄いかもしれない．試作問題としては良い題材だが，令和時代の問題としてはちょっと古臭いかな．それでは，昭和時代の私が，問1から解説していく．解説では，正解だけでなく，解答群にある誤選択肢（錯乱肢）についても触れる．

問1は，最小交換貨幣枚数と関数「**枚数**(金額)」の理解を問う問題である．この問1の内容が理解できれば，問2と問3は見通しよく解けるだろう．それでは，問題文を見てみよう．

S：46 円を支払うには，10 円玉 4 枚，5 円玉 1 枚，1 円玉 1 枚という 6 枚で払い方が最小の枚数になります。 A

T：そうですね。一方，同じ 46 円を支払うのに，51 円を支払って釣り銭 5 円を受け取る払い方では，支払いに 2 枚，釣り銭に 1 枚で，合計 3 枚の硬貨のやり取りになります。 B こうすると交換する硬貨の枚数の合計が最小になりますね。

S：これが上手な払い方ですね。

T：そうです。このように C 客と店が交換する硬貨の合計が最小となる枚数，すなわち「最小交換硬貨枚数」の計算を考えましょう。

下線Bが「上手な払い方」である．このような払い方を実生活でも実践していれば，「上手な払い方」の意味はすぐに分かるだろう．この上手な払い方を，下線Cのように定式化し，それを「最小交換硬貨枚数」と呼んでいる．

（ア）の正解は，下線Aに6枚と書いてあり，数字の6である．ここで，【関数の説明と例】を読んで，関数「**枚数**(46)」の戻り値は6になることを理解しておこう．

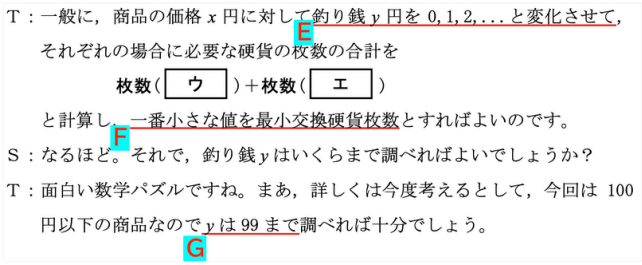
T：これは，例えば，**枚数**(46)＝［ア］と計算してくれるような関数です。これを使って最小交換硬貨枚数の計算を考えてみましょう。例えば，46 円支払うのに，51 円払って 5 円の釣り銭を受け取る払い方をした場合，客と店の間で交換される硬貨枚数の合計 D は，この関数を使うと，どのように計算できますか？

S：［イ］で求められますね。

（イ）について考える．下線Dより，客から店に支払う枚数が**枚数**(51)，店から客に支払う枚数が**枚数**(5)なので，（イ）の正解は⓪**枚数**(51)＋**枚数**(5)である．あるいは，〝客が51円支払う〟ので，**枚数**(46)の計算は不適であり，**枚数**(46)を含む解答群①と③は誤選択肢である．また枚数の合計を求めるので，減算を含む②も誤選択肢である．このように消去法で考えると，残りの⓪が正解になる．

正解を得るには，問題を解いて正攻法で正解を得る方法と，解答群から消去法で正解を得る方法がある．両方が使える場合は，正攻法で解いて，消去法で検算するのがよいだろう．

（ウ）と（エ）は，この問題の要である．



（イ）の正解が⓪**枚数**(51)＋**枚数**(5)なので，商品の価格が46円の場合は，51円支払って，5円の釣り銭を受け取ればよいことが分かる．商品の価格x円と釣り銭y円に，具体例を当てはめれば，x＝46，y＝5，x＋y＝51となる．したがって，（ウ）と（エ）の正解は，順不同で，①yと②x＋yである．選択肢②x＋yは客から店に支払う金額である．xが価格，yが釣り銭であることが分かれば，ウ・エの解答群にある価格xから釣り銭yを引くという③x−yは，誤選択肢らしいと分かる．

入試問題では，問1が，その後の問いを解くための前提条件となっていることが多い．このため，問1にあるT先生とSさんの会話も一字一句漏らさず理解しておこう．問1の問題文から分かることを整理する．要点は次の4つである．

1. 価格はx，釣り銭はy，客から店に支払う金額はx＋yである．
2. 下線Eと下線Gより，yは0から99まで調べる．
3. 硬貨枚数の合計は**枚数**(x＋y)＋**枚数**(y)で計算できる．
4. 下線Fの意味は，yを0から99まで変化させながら，**枚数**(x＋y)＋**枚数**(y)の一番小さな値を見つける，である．

問2について考える．（オ）と（カ）は，演算子「÷」と「%」の使い方を問う問題である．金額が46円で支払いに10円玉を使う場合，支払う10円玉の枚数は46÷10＝4枚，残金は46%10＝6円と計算できる．演算子「÷」を使うと枚数が求まり，演算子「%」を使うと金額が求まる．枚数を求める（オ）の正解は②46÷10，残金を求める（カ）の正解は③46%10である．オ・カの解答群の⓪と①は，見るからに “＋1” や “−1” が怪しいので，誤選択肢らしいと分かる．現状では，受験生にとって演算子「%」は馴染みが薄いかもしれないが，情報Iが浸透すれば，このような基本的な演算子の使い方を問う問題はなくなるだろう．

次の問題文と図1を見て，（キ）から（コ）について考える．

Sさんは，先生（T）との会話からヒントを得て，変数 **kingaku** に与えられた目標の金額（100 円以下）に対し，その金額ちょうどになる最小の硬貨枚数を計算するプログラムを考えてみた（図 1）。ここでは例として目標の金額を 46 円としている。

配列 **Kouka** に硬貨の額を低い順に設定している。なお，配列の添字は **0** から始まるものとする。最低額の硬貨が 1 円玉なので **Kouka[0]** の値は **1** となる。

先生（T）のヒントに従い，高額の硬貨から何枚まで使えるかを計算する方針で，**(4)**～**(6)** 行目のような繰返し文にした。この繰返しで，変数 **maisu** に支払いに使う硬貨の枚数の合計が計算され，変数 **nokori** に残りいくら支払えばよいか，という残金が計算される。

```
(1) Kouka = [1,5,10,50,100]
(2) kingaku = 46
(3) maisu = 0, nokori = kingaku
(4) i を [ キ ] ながら繰り返す：
(5)     maisu = [ ク ] + [ ケ ]
(6)     nokori = [ コ ]
(7) 表示する(maisu)
```

図1　目標の金額ちょうどになる最小の硬貨枚数を計算するプログラム

（キ）について考える．変数**i**がどのように使われるかが分かると簡単である．ケ・コの解答群を見ると，**Kouka[i]**って書いてある！　解答群にヒントがある場合がある．また，下線Hと下線Iより，**Kouka[i]**は100, 50, ..., 5, 1となればよい．すなわち，**Kouka[i]**は，**Kouka[4]**, **Kouka[3]**, ..., **Kouka[0]**と変化すればよい．（キ）の正解は①4から0まで1ずつ減らし，である．解答群の⓪と③は，配列の添え字が1から始まる場合には正解かもしれない選択肢である．現在の受験生が学ぶプログラミング言語の配列は，ほとんど（すべて？）0から始まる．このことより，受験生が⓪と③を選択することはないだろう．私（本解説の著者）としては，高額の貨幣から使うので，**Kouka = [100, 50, 10, 5, 1]**として，（キ）の正解は②0から4まで1ずつ増やし，とする方が好みである．これだとキの難易度は少し下がるだろう．

（ク）について考える．図1の(3)行目より，**maisu**の初期値は0である．下線Jより，図1のプログラムの(5)行目は，**maisu = maisu + ケ**という式になる．（ク）の正解は①**maisu**である．

（ケ）と（コ）について考える．解答群は同じなので，（ケ）と（コ）は同時に考えていこう．**maisu**は硬貨の枚数を保持し，**nokori**は残金を保持する．ここで，（オ）と（カ）の正解を思い出してほしい．演算子「÷」を使うと枚数が求まり，演算子「%」を使うと残金が求まる．図1の(5)行目は**maisu = maisu + ケ**より，（ケ）は額面が**Kouka[i]**の硬貨の枚数である．すなわち（ケ）の正解は⓪**nokori ÷ Kouka[i]** である．

図1の(3)行目を見ると，**nokori = kingaku**とある．**nokori**は，初期値が**kingaku**であり，繰り返すごとに，額面**Kouka[i]**の硬貨で支払ったあとの残金となる．残金を求めるには演算子「%」を使えばよいので，（コ）の正解は①**nokori % Kouka[i]**である．

ここで，演算子「÷」と「%」について復習しておこう．**nokori**の値より**Kouka[i]**の方が大きい場合，**nokori ÷ Kouka[i]**の値は0となり，**nokori % Kouka[i]**の値は**nokori**と同じになる．これは，図1

の(5)と(6)行目で，**nokori**の値より**Kouka[i]**の方が大きい場合，**maisu**は増えず**nokori**は減らないこと，すなわち**Kouka[i]**に対応する硬貨は使わない，ことを意味する．

ケ・コの解答群を見てみよう．解答群の②と③は枚数**maisu**を硬貨1枚当たりの金額**Kouka[i]**で割っているので，枚数÷金額／枚数すなわち「枚数を1枚当たりの金額で割った数」となる．これが金額であるとは思えないので，②と③は誤選択肢らしいと分かる．解答群の⓪と①は，金額÷金額／枚数すなわち「金額を1枚当たりの金額で割った数」なので，これはなんらかの枚数であると推測できる．このように，（ケ）と（コ）については，解答群を見ただけで，正解の見当がつけられる．

さて，図1のプログラムは，最小の硬貨枚数を計算すると書いてあるが，硬貨の額面が異なる場合でも必ずそうなるだろうか．たとえば，**Kouka = [1, 5, 20, 40, 50, 100]**で，金額が80円だとする．図1のプログラムは高額硬貨から優先して支払うので，金額80円の場合は，50円玉1枚＋20円玉1枚＋5円玉2枚で，枚数は4枚になる．これは最小の硬貨枚数ではない．最小の硬貨枚数は，40円玉2枚で，枚数は2枚である．図1のプログラムが，最小の硬貨枚数を計算できるかどうかは，配列**Kouka**の値と関係している．高額貨幣から優先して支払う方法は，いつも最小の硬貨枚数が計算できるわけではない．しかし，現実社会では，20円玉や40円玉はないので，図1のプログラムで最小の硬貨枚数が計算できる．安心してください．

問3について考える．

T：プログラム（図１）ができたようですね。それを使えば，関数「**枚数**(金額)」のプログラムができます。関数の引数として与えられる金額の値をプログラム（図１）の変数 **kingaku** に設定し，**(7)** 行目の代わりに変数 **maisu** の値を関数の戻り値とすれば，関数「**枚数**(金額)」のプログラムとなります。では，その関数を使って最小交換硬貨枚数を計算するプログラムを作ってみましょう。ここでも，100 円以下の買い物として考えてみます。

T先生は，図1のプログラムを使って，関数「**枚数**(金額)」のプログラムができるといっている．しかし，下線Kについては，参考文献[1) 2)]を見ても，具体的な書き方は分からない．関数の戻り値の書き方は，必要に応じて公開されるだろう．また，図1のプログラムで使われた**maisu**，**nokori**，**Kouka**については，問3では使われない．これらの変数は，関数「**枚数**(金額)」の中だけで使われる変数である．

問3では，関数「**枚数**(金額)」が使える前提で解けばよい．それでは，問3の問題文をよく読むことにしよう．

Sさんは，図2のようなプログラムを作成した。変数 **kakaku** に与えられる商品の価格に対して，釣り銭を表す変数 **tsuri** を用意し，妥当な **tsuri** のすべての値に対して交換する硬貨の枚数を調べ，その最小値を求めるプログラムである。なお，ここでは例として商品の価格を 46 円としている。

このプログラムでは，先生（T）のアドバイスに従い，釣り銭無しの場合も含め，99 円までのすべての釣り銭に対し，その釣り銭になるように支払う場合に交換される硬貨の枚数を求め，その最小値を最小交換硬貨枚数として計算している。

最小値の計算では，これまでの払い方での最小枚数を変数 **min_maisu** に記憶しておき，それより少ない枚数の払い方が出るたびに更新している。**min_maisu** の初期値には，十分に大きな値として 100 を用いている。100 円以下の買い物では，使う硬貨の枚数は 100 枚を超えないからである。

まず，問1の4つの要点を思い出そう．この問題文から，商品の価格xが**kakaku**に，釣り銭yが**tsuri**に対応することが分かる．また，下線Lは，問1の下線E，FとGに対応する．そして，問1の2番目の要点 “yは0から99まで調べる” は，**tsuri**は0から99まで調べる，と読み換えることができる．それでは，図2の（サ）から（タ）について考えていこう．

```
(1) kakaku = 46
(2) min_maisu = 100
(3) [ サ ] を [ シ ] から 99 まで 1 ずつ増やしながら繰り返す：
(4)     shiharai = kakaku + tsuri
(5)     maisu = [ ス ] + [ セ ]
(6)     もし [ ソ ] < min_maisu ならば：
(7)         [ タ ] = [ ソ ]
(8) 表示する(min_maisu)
```

図2　最小交換硬貨枚数を求めるプログラム

（サ）と（シ）について考える．99まで1ずつ増やす変数は，**tsuri**であり，初期値は**0**である．（サ）の正解は③**tsuri**であり，（シ）の正解は⓪**0**である．

（ス）と（セ）について考える．これらは，問1の4番目の要点である “**枚数**(x＋y)＋**枚数**(y)” に対応する．“**枚数**(x＋y)” は**枚数**(**kakaku + tsuri**)となるが，これは解答群にはない．図2の(4)行目を見ると，**shiharai = kakakku + tsuri**となっている．（ス）と（セ）の正解は，順不同で，⓪**枚数**(**shiharai**)と②**枚数**(**tsuri**)である．

（ソ）と（タ）について考える．下線Mより，(5)行目で計算した**maisu**が**min_maisu**より小さい場合，**min_maisu**を**maisu**で更新すればよい．（ソ）の正解は⓪**maisu**であり，（タ）の正解は①**min_maisu**である．

さて，図2のプログラムの(6)行目を，**maisu ≦ min_maisu** に変更したら，どうなるだろうか．この変更でも最小交換硬貨枚数は正しく求まる．**Kouka = [1,5,10,50,100]**では，この変更の影響はない．もし，10円玉と50円玉の代わりに30円玉と40円玉があるとすると，**Kouka = [1, 5, 30, 40, 100]**となる．この**Kouka**で，金額75円の最小交換硬貨枚数を求めてみる．最小交換硬貨枚数は3枚である．そうなる払い方は，**枚数**(75)＋**枚数**(0)，**枚数**(80)＋**枚数**(5)，**枚数**(105)＋**枚数**(30)の3通りである．(6)行目を，**maisu < min_maisu** とした場合は**枚数**(75)＋**枚数**(0)が求まり，**maisu ≦ min_maisu** とし

た場合は**枚数**(105)＋**枚数**(30)が求まる．現実的には，30円玉と40円玉はないので，このようなことはないが，プログラム中に大小比較があったときは，こんな風にいろいろ検討してみるとよい．

プログラムを書くと，そのプログラムの実行時間が気になるだろう．図1のプログラムの実行時間は，実行時間の大半を占める繰り返し部分だけに注目し，(4)行目からの繰り返し回数と比例すると考えられる．繰り返す回数は，**kingaku**の値に関係なく，配列**Kouka**の要素数と同じ5回になる．

図2のプログラムの実行時間はどうだろうか．注目するのは，図2の(3)行目からの繰り返し回数である．商品の価格が100円以下だと，0から99までなので100回になる．商品の価格が1,000円以下だと，(3)行目を，**tsuri**を0から999まで1ずつ増やしながら繰り返す，と変更するので，0から999までの1,000回になる．商品の価格に応じて，繰り返す回数が増えるので，実行時間も長くなる．図2のプログラムは，払い枚数と釣り銭枚数の組合せを求めながら，その中から最小交換硬貨枚数を見つけている．図2のプログラム（アルゴリズム）は，図1のプログラムと比べて，効率が悪いといえる．ここでは，実行時間の代わりに繰り返し回数でプログラム（アルゴリズム）の効率を評価してみた．このような評価を計算量という．

第3問は，「最小交換硬貨枚数」について，問1で定式化し，問3で，そのプログラムを扱っている．問2は，問3の2カ所で使われる関数「**枚数**(金額)」を扱っている．問1から問3までの流れは，受験生にとって，解きやすい流れである．また，問題文を注意深く読んでから解答群を見れば，誤選択肢の見当がつく．また，解答の解説だけでなく，**Kouka**の値を替えてみたり，<を≦に変更してみたりと，ちょっとプログラムで遊んでみた．理解を深めるには，この手の遊びは有効だと思う．最後に，第3問に興味を持った読者は，貪欲法，整数計画法，最適化や計算量などについて調べるとよいだろう．

参考文献

1）水野修治：令和7年度大学入学共通テスト『情報I』の実施に向けて～問題作成方針に関する検討の方向性と試作問題～，情報処理，Vol.64, No.2, pp.74-77 (2023).
https://doi.org/10.20729/00223448
2）大学入試センター：令和7年度試験の問題作成の方向性，試作問題等
https://www.dnc.ac.jp/kyotsu/shiken_jouhou/r7/r7_kentoujoukyou/r7mondai.html

（2024年1月26日受付）
（2024年2月21日note公開）

■冨澤眞樹（正会員）
東京農工大学大学院工学研究科電子情報工学専攻博士後期課程修了，博士（工学）．情報システムの開発と，少しだけ計算量とグラフアルゴリズムに興味を持つ．

情報処理学会ジュニア会員へのお誘い

小中高校生，高専生本科～専攻科1年，大学学部１～3年生の皆さんは，情報処理学会に**無料で入会**で

きます．**会員になると有料記事の閲覧，情報処理を学べるさまざまなイベントにお得に参加できる等のメリットがあります**．ぜひ，入会をご検討ください．入会は**こちら**から！