平成25年度 茨城大学大学院 理工学研究科
博士前期課程 入学試験問題

情報工学専攻

アルゴリズムとプログラミング

注意事項

1. 試験開始の合図があるまで，この問題冊子の中を見てはいけません．
2. この試験に使用する解答用紙は 7枚です．枚数を確認して，解答用紙の表題の右側にある（ ）内に,（その1）から（その7）までを記入してください．
   問題1の問1に対し（その1）を，
   問題1の問2，問3，問4，問5に対し（その2）を，
   問題2の問1に対し（その3）を，
   問題2の問2に対し（その4）を，
   問題2の問3に対し（その5）を，
   問題2の問4に対し（その6）を，
   問題2の問5に対し（その7）を使用してください．
3. 解答用紙の全てのページに，解答科目名と受験番号を記入してください．
   氏名欄は記入しないでください．受験番号に記入漏れや記入間違いがあると，試験は無効になりますので，手元の受験票で確認のうえ注意深く記入してください．
   【上記の2と3に記載の事項は，試験開始前に必ず記入してください.】
4. 試験中に問題冊子の印刷不鮮明，ページの落丁・乱丁等に気が付いた場合は，手を上げて試験監督に知らせてください．
5. 携帯電話の電源は必ず切ってカバン等に入れ，身に着けないでください．また，携帯電話を時計として使用することはできません．
6. 試験中に辞書類・電卓を使用することはできません．

英語対応表

ナップサック問題	knapsack problem
容量	capacity
重さ	weight
価値	value
品物	item
超える	exceed
選ぶ	select
合計	total sum
最大値	the maximum value
番号	number
整数	integer
漸化式	recurrence relation
解	solution
再帰呼び出し	recursive call
2次元配列	two-dimensional array
試験結果	result of examination
記録する	record
テキストファイル	text file
行	line
学生番号	student id number
氏名	name
得点	score
コンマ	comma
順番	order
長さ	length
文字	character
英数字	alphanumeric character
空白	space
バイト	byte
整列	sort
構造体	structure
線形リスト	linear list
先頭ノード	first node
参照する	reference
関数	function
文字列	character string
動的メモリ	dynamic memory
割り当てる	allocate
標準ライブラリ	standard library
変換する	convert
末尾ノード	last node
追加する	add
降順	descending order

アルゴリズムとプログラミング

問題 1　ナップサック問題とは，容量 U のナップサックが 1 つと重さと価値の決まった n 個の品物が与えられ，重さの合計が U を超えないように品物を選んだとき，価値の合計の最大値はいくつになるかを求める問題である．n 個の品物には 1 番から n 番まで番号が付いてあり，i 番目の品物の重さ及び価値を，それぞれ，w_i 及び v_i と書くことにする．重さと価値は，共に，0 以上の整数であるとする．例えば，下の表のように 5 個の品物が与えられたする．

品物 (i)	1	2	3	4	5
重さ (w_i)	5	7	8	12	10
価値 (v_i)	25	45	50	65	60

もし，$U = 20$ が与えられたとすると，価値の合計は，$\{1,2,3\}$ の 3 つの品物を選んだとき最大になり，その値は 120 である．

1 番目から i 番目までの品物から，重さの合計が u を超えないように選んだときの価値の合計の最大値を $k(i,u)$ と書くことにする（ただし $1 \leq i \leq n$ 及び $u \geq 0$ である）．すると，以下の漸化式が成り立つ．

$$k(1,u) = \begin{cases} 0 & (u < w_1) \\ v_1 & (u \geq w_1) \end{cases}$$

$$k(i,u) = \begin{cases} k(i-1,u) & (u < w_i \text{ または } k(i-1,u) > k(i-1,u-w_i)+v_i) \\ k(i-1,u-w_i)+v_i & (\text{それ以外の場合}) \end{cases}$$

$k(n,U)$ が問題の解となる．この漸化式を利用して，ナップサック問題を解く C 言語のプログラムを作成することを考える．ここでは再帰呼び出しを使わずに，整数型の 2 次元配列 D を用意し，$D[i][u] = k(i,u)$ となるように配列を埋めていくプログラムを作成する．

リスト 1 に，上記の表の品物に対し，ナップサックの容量 U が与えられたときに，価値の合計の最大値を求める C 言語のプログラムの一部を示す．その下に，ナップサックの容量として 20 を入力したときの実行例を示す．このプログラムについて，以下の問 1 ～問 5 に答えよ．

問 1　リスト 1 のプログラム中の，整数型の 2 次元配列 D を埋めていく部分を記述せよ．
（ヒント）配列は $D[1][0]$ から順に $D[n][U]$ まで埋めればよい．

問 2　30 が入力されたとき，プログラムの出力を書け．

問 3　30 が入力されたとき，配列要素 $D[1][30]$ の値はいくつになるか答えよ．

問 4　40 が入力されたとき，配列要素 $D[5][20]$ の値はいくつになるか答えよ．

問 5　リスト 1 の 8 行目から 10 行目を以下に置き換える．50 が入力されたとき，配列要素 $D[10][40]$ の値はいくつになるか答えよ．

```
int n=10; /* 品物の個数 */
int w[11]={0,  2,  4,  6,  8, 10, 12, 14, 16, 18, 20}; /* 品物の重さ */
int v[11]={0,  5,  8, 10, 15, 18, 20, 30, 35, 50, 40}; /* 品物の価値 */
```

リスト１：ナップサック問題を解くプログラム

```
#include <stdio.h>

#define max_U 10000
#define max_n 100

int D[max_n+1][max_U+1];

int n=5; /* 品物の個数 */
int w[6]={0,  5,  7,  8, 12, 10}; /* 品物の重さ */
int v[6]={0, 25, 45, 50, 65, 60}; /* 品物の価値 */

int main(void){
    int i, u;
    int U;

    scanf("%d", &U);

    ┌──────────────────────────────┐
    │                              │
    │            問１              │
    │                              │
    └──────────────────────────────┘

    printf("%d\n", D[n][U]);

    return 0;
}
```

●実行例

```
    > a.out
    20
    120
    >
```

問題2

図1は学生の試験結果を記録したテキストファイルの例である．各行は学生番号，氏名，得点がコンマ (,) で区切られて順番に記録されている．各行の長さは 255 文字以内，学生番号，氏名の長さはそれぞれ 15 文字，127 文字以内とする．また，学生番号，氏名，得点はコンマを含まないものとし，得点は整数の文字列表現とする．なお，使用される文字は英数字，空白及びコンマのみで，1文字=1バイトと仮定してよい．

```
A1011,Ibaraki Taro,70
A1012,Hitachi Hanako,93
A1013,Mito Umeko,85
A1014,Ami Ichiro,78
```

図 1: 学生の試験結果を記録したテキストファイルの例

このテキストファイルを読み込んで，得点順に整列するプログラムを C 言語で作成する．リスト2にプログラムの一部を示す．このプログラム内では，1行分の試験結果を RECORD 型構造体に格納し，LIST 型構造体による線形リストを使ってそれらの順番を管理するものとする．なお，プログラムを単純にするために，線形リストの先頭ノードは RECORD 型データへのポインタを使用しないダミーとし，最初の RECORD 型データは線形リストの2番目のノードから参照されるものとする．その構造を図2に図示する．

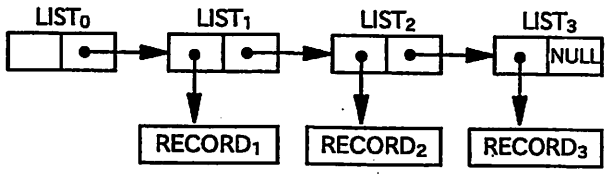

図 2: 線形リストの構造

以下の問1～問5に答えよ．

問1 関数 RECORD *parse(char *line) は1行分の文字列から，学生番号，氏名，得点の値を取り出して，新しい RECORD 型データを作成する関数である．この関数の中身を記述せよ．学生番号，氏名の長さが制限を越えた場合には NULL を返すこと．なお，動的メモリの割り当てには標準ライブラリの関数 void *malloc(size_t size) を用い，文字列を整数へ変換するには int atoi(const char *str) を用いること．標準ライブラリの関数

```
char *strtok(char *str, const char *delimiter)
char *strcpy(char *dest, const char *src)
```

の使用は任意とする．リスト2に string.h のインクルードは記述されていないが，記述されているとみなしてよい．

問2 関数 void add(LIST *list, RECORD *new_record) は，線形リストの末尾ノードの次に RECORD 型データを追加する関数である．この関数の中身を記述せよ．線形リストの新しいノードの作成には，リスト2で定義されている関数 createlist() を用いること．

問3 関数 int readall(LIST *list, const char *filename) は指定されたファイル名のファイルからすべてのデータを1行ずつ読み込み，読み込んだ順番に線形リストに追加する関数である．ファイルのオープンに成功した場合には SUCCESS，失敗した場合には ERROR を返

す．この関数の中身を記述せよ．なお，RECORD 型データの作成には問 1 の関数 parse()，線形リストへのデータの追加には問 2 の関数 add() を用いること．さらに，ファイルの操作には，標準ライブラリの関数

```
FILE *fopen(const char *filename, const char *mode)
char *fgets(char *str, int size, FILE *stream)
int fclose(FILE *stream)
```

を用いること．

問 4 線形リストが得点の降順になるように整列を行う関数 `void sort(LIST *list)` の中身を記述せよ．整列アルゴリズムの選択は自由とする．

問 5 問 1 から問 4 が正しく記述されたときの，リスト 2 のプログラムの出力結果を示せ．テキストファイル `test.txt` の内容は図 1 のとおりとする．

リスト 2: 学生の試験結果を得点順に整列するプログラム

```
#include <stdio.h>
#include <stdlib.h>

#define SUCCESS 1
#define ERROR 0

#define MAX_LENGTH_LINE 255 /* 読み込むテキストファイルの 1 行の最大文字数 */
#define MAX_LENGTH_ID 15 /* 学生番号の最大文字数 */
#define MAX_LENGTH_NAME 127 /* 氏名の最大文字数 */

typedef struct { /* 1 行分の試験結果を格納する構造体 */
  char id[MAX_LENGTH_ID+1];  /* 学生番号 */
  char name[MAX_LENGTH_NAME+1]; /* 氏名 */
  int score; /* 得点 */
} RECORD;

typedef struct list { /* 線形リストのノードを構成する構造体 */
  RECORD *value; /* RECORD 型データへのポインタ */
  struct list *next; /* 次のノードへのポインタ */
} LIST;

LIST *createlist(){
  LIST *result;

  result = (LIST *)malloc((size_t)sizeof(LIST));
  result->value = NULL;
  result->next = NULL;

  return result;
}
```

次ページへ続く

リスト 2: 学生の試験結果を得点順に整列するプログラム（続き）

```
RECORD *parse(char *line){  問1  }

void add(LIST *list, RECORD *new_record){  問2  }

int readall(LIST *list, const char *filename){  問3  }

void sort(LIST *list){  問4  }

void printall(LIST *list){
  LIST *current;
  RECORD *record;

  current = list;
  if(current->next != NULL){
    current = current->next;
    record = current->value;
    printf("[%s][%s][%d]\n",record->id, record->name, record->score);
    printall(current);
  }
}

int main(void){
  LIST *list;

  list = createlist(); /* 線形リストの先頭ノード (ダミー) の作成 */

  if(readall(list, "test.txt")==ERROR){
     printf("ファイルの読み込みに失敗しました\n");
     return -1;
  }

  sort(list);

  printall(list);

  return 0;
}
```