【 前 期 募 集 】

令和 4 年度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

入　学　試　験　問　題

No. 1

コース等	コンピュータ理工学コース	試験分野	専門科目

試験時間は 1 時間 30 分です．試験監督から指示があるまで，この表紙をめくってはいけません．
次ページ以降に問題が，右上に番号付けされた用紙に分けて出題されています．配点は 140 点です．解答にあたっては，解答用紙の表紙の指示に従いなさい．

解答開始の合図の後，各分野の問題について下表中に示す No. の用紙が綴じ込まれていることを確認しなさい．用紙に乱丁・落丁がある場合には，手を挙げて試験監督に知らせなさい．

	分野名	問題用紙のページ番号
必須	アルゴリズムとデータ構造，並びにプログラミング	No. 2~10

解答は，原則，1 問につき 1 枚を使用する．もしも解答用紙のスペースが不足の場合には，手を挙げて試験監督に知らせること．

すべての解答用紙について，受験番号欄に受験番号を記入の上で試験終了後に提出しなさい．本用紙を含むすべての問題用紙についても，回収します．

【 前 期 募 集 】

令 和 4 年 度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

入　学　試　験　問　題

No　2

コース等	コンピュータ理工学コース	試験分野	アルゴリズムとデータ構造，並びにプログラミング

問 1 以下の設問に答えよ．

(1) 逆ポーランド記法 (Reverse Polish notation; RPN) は，演算子（オペレータ）を被演算子（オペランド）の前に記述する記法であり，後置記法とも呼ばれる．例えば，2 と 3 を足して，それに 4 を掛けるという式は，通常の中置記法では $(2+3)*4$ と表記され，逆ポーランド記法では 2 3 + 4 * となる．

(a) 中置記法の式 $2+3*4$ を RPN で記述せよ．

(b) 中置記法の式 $(8/(6-4))*2$ を RPN で記述せよ．

(c) RPN の式 3 4 + 5 − 6 * を中置記法で記述せよ．

(d) RPN の式 7 6 5 4 * 3 2 + / − * を中置記法で記述せよ．

令和４年度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

入　学　試　験　問　題

No　3

コース等	コンピュータ理工学コース	試験分野	アルゴリズムとデータ構造，並びにプログラミング

(2) Stack を double 型の値を保持するスタックを表すクラスとする．このクラスのコンストラクタとメンバ関数を以下に示す．

関数名	説明
`Stack()`	空のスタックを生成するコンストラクタ
`bool empty()`	スタックが空ならば `true` を，それ以外の場合は `false` を返す
`void push(double x)`	スタックに要素 `x` を追加する
`double pop()`	スタックに最後に追加された要素を削除し，その要素を返す

Stack オブジェクトへの参照を受け取り，そのスタックの内容を画面に出力する関数 void print_stack(Stack& s) の定義を示せ．出力形式は，以下のサンプルプログラムの実行結果に従うものとする．必要であれば追加の関数を定義してよい．ただし大域変数および静的局所変数の利用は認めない．

サンプルプログラム（sample program）

```
int main(void) {
  Stack s;

  s.push(1.2);
  s.push(3.4);
  s.push(5.6);
  print_stack(s);
  cout << "------" << endl;

  s.pop();
  print_stack(s);
  cout << "------" << endl;

  s.pop();
  s.pop();
  print_stack(s);

  cout << "------" << endl;

  return 0;
}
```

実行結果（execution results）

```
5.6
3.4
1.2
------
3.4
1.2
------
------
```

令和４年度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

入　学　試　験　問　題

No　4

コース等	コンピュータ理工学コース	試験分野	アルゴリズムとデータ構造，並びにプログラミング

(3) RPN で記述された式はスタックを利用することで評価可能である．ソースコード１は RPN で記述された式を評価するプログラムである．(2) で示した Stack クラスおよび print_stack 関数がソースコード中で利用可能と仮定する．このプログラムを実行したときの出力結果を示せ．

ソースコード１

```
#include <iostream>
#include <sstream>
#include <map>

using namespace std;

class Operator {
   public:
    virtual void calc(Stack& stk) = 0;
};

class PlusOperator : public Operator {
   public:
    void calc(Stack& stk) {
        double op2 = stk.pop();
        double op1 = stk.pop();
        stk.push(op1 + op2);
    }
};

class MultiplyOperator : public Operator {
   public:
    void calc(Stack& stk) {
        double op2 = stk.pop();
        double op1 = stk.pop();
        stk.push(op1 * op2);
    }
};

// Returns true if the string s represents a double number, false otherwise
bool isDouble(string& s) {
  istringstream i(s);
  double temp;
  return (i >> temp) ? true : false;
}

                                                    // Continued to the next page
```

令 和 4 年 度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

入　学　試　験　問　題

No　5

コース等	コンピュータ理工学コース	試 験 分 野	アルゴリズムとデータ構造, 並びにプログラミング

```
class RPNCalculator {
   private:
    Stack stk;
    map<string, Operator*> ops;

   public:
    RPNCalculator() {
        ops["+"]  = new PlusOperator();
        ops["*"]  = new MultiplyOperator();
        ops["-"]  = new MinusOperator();     // defined in (5)
        ops["sp"] = new SquareOperator();    // defined in (6)
    }

    void add(string s) {
        if (isDouble(s)) {
            // Convert the string s to a double number and add it to the stack
            stk.push(stod(s));
            return;
        }
        if (ops.count(s) == 0) { // If the operator s is not defined
            cout << "Error: operator " << s << " is not supported" << endl;
            return;
        }
        ops[s]->calc(stk);
    }

    void print() {
        cout << "---" << endl;
        print_stack(stk);
    }
};

int main(void) {
    string input[] = { "10", "20", "30", "+", "*" };
    int num = sizeof(input)/sizeof(string); // the number of elements in input

    RPNCalculator rpn;
    for (int i=0; i < num; i++) {
        rpn.add(input[i]);
        rpn.print();
    }

    return 0;
}
```

【 前 期 募 集 】

令 和 4 年 度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

入　学　試　験　問　題

No　6

コース等	コンピュータ理工学コース	試験分野	アルゴリズムとデータ構造，並びにプログラミング

(4) スタックを利用することで，RPN で記述された式をなぜ評価できるのか説明しなさい．

(5) 引き算を含む式を評価するため，`PlusOperator` クラスと同様にして，`MinusOperator` クラスを定義せよ．

(6) `sq` を 2 乗を求める演算子とする．例えば，4 を 2 乗し，そこから 3 を引く式は，RPN 形式で `4 sq 3 -` と表すことができる．2 乗演算子を表すクラス `SquareOperator` クラスの定義を示せ．

(7) `Operator` は抽象クラスである．抽象クラスとその利点について説明しなさい．

【 前期募集 】

令和４年度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

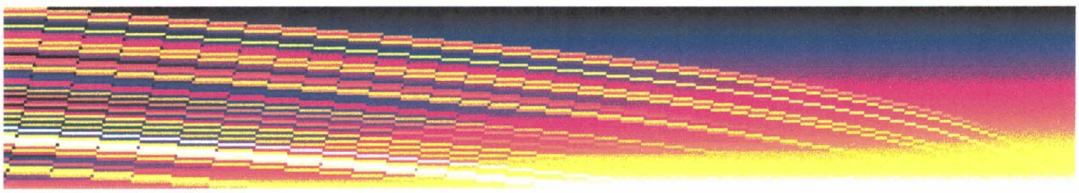
入　学　試　験　問　題

No　7

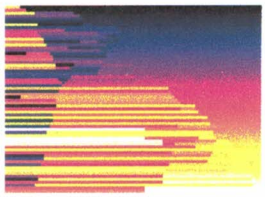
コース等	コンピュータ理工学コース	試験分野	アルゴリズムとデータ構造，並びにプログラミング

問２　図 A，B，C はソート処理によってデータ配列が並べ替えられていく過程を可視化したものである．データ配列は縦に並べてあり，データの大小関係は色凡例に従っている．横軸に沿って並べ替えが行われていき，一番右の列がソートした結果である．以下の設問に答えなさい．

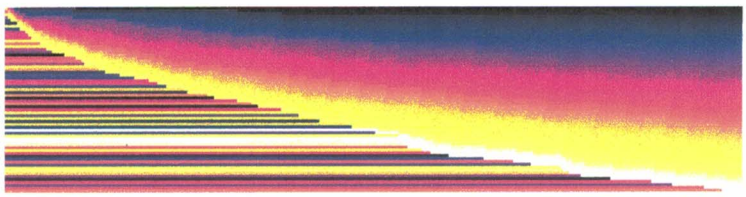
A



B



C



色凡例

小さい　　大きい

(1) 図 A，B，C はそれぞれ以下のどのソートアルゴリズムを可視化したものであるか．
(a) バブルソート
(b) 挿入ソート
(c) クイックソート
(d) マージソート
(e) 選択ソート

(2) 設問（1）の回答の理由を述べなさい．

(3) 上記の５つのアルゴリズムを計算量の観点から二つのグループに分類し，その根拠を説明しなさい．

(4) クイックソートに，挿入ソートを組み合わせて使用すると処理をさらに高速にできる手法が知られている．その手法を説明しなさい．

令和４年度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

入　学　試　験　問　題

No　8

コース等	コンピュータ理工学コース	試験分野	アルゴリズムとデータ構造，並びにプログラミング

問３　ハッシュ関数を用いた検索に関する以下の問いに答えなさい．

(1)　検索に用いられるハッシュ関数に求められる性質について説明しなさい．

(2) 文字列 *key* に対する次の 3 つのハッシュ関数の性能について説明しなさい．

以下では，ハッシュ表のサイズ（登録できるデータ数）を $t\_size = 2^{24}$ とし，*key* は長さ length(*key*)の文字型配列と仮定する．*key* の各文字は *key*[i]で取り出せ，ord 関数で文字コードを整数値として出力できるものとする．また *hash* は 32bit の符号無し整数変数とする．

```
1: global t_size ← 2^24;

2: function HASH1(key)
3:     i ← 0;  hash ← 0
4:     for i < length(key) do
5:         hash ← hash + ord(key[i]);          ▷ hash は 32bit の符号無し整数
6:         i ← i + 1
7:     end for
8:     return (hash mod t_size)
9: end function

10: function HASH2(key)
11:     i ← 0;  hash ← 0
12:     for i < length(key) do
13:         hash ← hash × 2^6 + ord(key[i]);    ▷ hash は 32bit の符号無し整数
14:         i ← i + 1
15:     end for
16:     return (hash mod t_size)
17: end function

18: function HASH3(key)
19:     i ← 0;  hash ← 0
20:     for i < length(key) do
21:         hash ← hash × 61 + ord(key[i]);     ▷ hash は 32bit の符号無し整数
22:         i ← i + 1
23:     end for
24:     return (hash mod t_size)
25: end function
```

令 和 4 年 度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

入　学　試　験　問　題

No　9

コース等	コンピュータ理工学コース	試験分野	アルゴリズムとデータ構造，並びにプログラミング

(3) ハッシュ法の代表的な手法として分離チェイン法と開番地法がある．それぞれの概要を説明し，その長所と短所を比較しなさい．

(4) 以下は開番地法に基づくハッシュ法を実現する擬似コードである．このコードは，データの追加 INSERT と削除 DELETE を何回か行った後に検索 FIND や DELETE を行うと，正しく動かない場合がある．

(a) 正しく動かない追加，削除，検索の操作列の例を示しなさい．併せて，その動作不良の原因を説明しなさい．ハッシュ関数 HASH は適当なものを仮定して説明すること．

(b) 正しく動作させるためのコードの修正法について説明しなさい．

```
1: global t_size ← 2^24;
2: global h_table ← new hash_table[t_size];
3:       ▷ 各 h_table[i] は，検索キー key，値 value，状態 stat の 3 つの要素からなる構造体

4: i ← 0
5: for i < t_size do
6:     h_table[i].stat ← EMPTY ;                    ▷ ハッシュ表 h_table の初期化
7:     i ← i + 1
8: end for

9: procedure INSERT(key, val)
10:    i ← HASH(key)
11:    while h_table[i].stat = ACTIVE do
12:        i ← (i + 1) mod t_size
13:    end while
14:    h_table[i].key ← key;
15:    h_table[i].val ← val;
16:    h_table[i].stat ← ACTIVE
17: end procedure
```

令 和 4 年 度
山梨大学　大学院医工農学総合教育部　修士課程 工学専攻

入　学　試　験　問　題

No　10

コース等	コンピュータ理工学コース	試験分野	アルゴリズムとデータ構造，並びにプログラミング

```
18: function FIND(key)
19:     i ← HASH(key)
20:     while h_table[i].stat ≠ EMPTY do
21:         if h_table[i].key = key then
22:             return h_table[i].val
23:         else
24:             i ← (i + 1) mod t_size
25:         end if
26:     end while
27:     return Not_found
28: end function

29: function DELETE(key)
30:     i ← HASH(key)
31:     while h_table[i].stat ≠ EMPTY do
32:         if h_table[i].key = key then
33:             h_table[i].stat ← EMPTY ;
34:             return Succeed
35:         else
36:             i ← (i + 1) mod t_size
37:         end if
38:     end while
39:     return Not_found
40: end function
```

(5) 手続き INSERT では *h_table* 表の空き領域の管理を行っていない．このために生じる不都合を説明しなさい．また INSERT の修正法についても説明しなさい．

(6) 開番地法では，ハッシュ表の半分程度が埋まると，急速にハッシュ関数の衝突が増加することが知られている．ハッシュ表が半分程度埋まった場合，ハッシュ表の大きさを 2 倍程度に拡大する対応がよく行われるが，その際に注意すべきことを説明しなさい．

【 前 期 募 集 】

令和４年度
山梨大学 大学院医工農学総合教育部

修士課程（工学専攻） 前 期 募 集

受験番号	

入 学 試 験 解 答 用 紙

コース等	コンピュータ理工学コース		
試験分野	アルゴリズムとデータ構造, 並びにプログラミング	採　　点	

問（　　） 解　答　　（注意：各問について各１枚の解答用紙を使用すること。）

※ 裏面も使用するときは、☑を付して下さい。　　□ 裏面あり