【 前 期 募 集 】

令和６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. 2~12

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 6 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	試験科目	アルゴリズムとデータ構造，並びにプログラミング

問 1 スタックオーバーフローとは何か，また，スタックオーバーフローが発生する処理の例を，合わせて 100 字程度で答えなさい．

問 2 コンピュータを使い，以下の公式を用いて，関数 f の x における微分値 $y' = f'(x)$ を数値的に計算する．プログラムにおいて x，h，$f(x)$，$f(x+h)$ 等の値は浮動小数点数で表現する．y'を求めるうえで，浮動小数点数を用いて数値計算することに起因する複数の問題が起きる可能性がある．起きる可能性のある問題のうち 2 つを選び，合わせて 50～150 字程度で説明しなさい．それらが以下の式のどの部分を原因としてどのような仕組みで起きるのか，具体的に述べること．

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}$$

【 前 期 募 集 】

令 和 6 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	試験科目	アルゴリズムとデータ構造，並びにプログラミング

問 3 互いに異なる N個の整数 $a_0, a_1, a_2, \ldots, a_{N-1}$ が整数型の配列 a[]に格納されている．いま，自然数 k，$0 \leq k \leq N-1$ が与えられたとき，a_kの順位，すなわち，a_kがこれら N個の整数のなかで何番目に小さいか，を求める順位探索関数

```
int findRank(int a[], int N, int k)
```

を考える．方針として，

1. ソート：配列 a[]を小さい順（昇順）に整列（ソート）した配列を b[] とする．
2. 探索：k が与えられたとき，要素 a[k] と値が一致する要素 b[j]の添え字 j を探索する．j は配列 a[]における a[k]の順位であり，これを戻り値として返す．

の手順で処理を行うとする．

(1) 初めに，ソートについて考える．以下のプログラムの関数 insertionSort と関数 quickSort は，整数を要素とする配列 a[]を小さい順（昇順）にソートする．用いるアルゴリズムは insertionSort は単純挿入ソート，quickSort はクイックソートである．プログラムの空欄 A と B を埋めなさい．複数行のコードを回答しても良い．

(2) 関数 insertionSort，関数 quickSort のそれぞれについて，a[]の要素数を Nとするときの，平均の時間計算量および最悪の時間計算量を $O(\ )$記法で記しなさい．これらがそれぞれ安定（stable）なソートか否かも記すこと．ここで，ソートが安定であるとは，同一の値を持つ要素間の順序が，ソートの前後で不変なことを指す．

(3) 以下の表は，上記の関数 insertionSort と関数 quickSort を，$N = 6$の場合について表に示す複数の a[N]の内容で実行した結果である．このとき，単純挿入ソート insertionSort 中の関数 swap(a[j],a[j-1])，およびクイックソート quicksort から呼ばれる関数 partition 中の関数 swap(a[i],a[j])の実行回数が以下の表の通りになった．実行回数が空欄である C ～ F について，それぞれの関数 swap の実行回数を記入しなさい．

a[]	関数 insertionSort swap(a[j],a[j-1])	関数 quicksort swap(a[i],a[j])
{3,9,4,6,1,5}	C	D
{1,3,4,5,6,9}	E	F
{9,6,5,4,3,1}	15	6
{1,3,4,9,6,5}	3	7

【 前 期 募 集 】

令和６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	試験科目	アルゴリズムとデータ構造，並びにプログラミング

(4) 平均の時間計算量が低くなるように関数 quickSort のふるまいを改善したい．様々な改善策が考えられるが，もし関数 partition を書き換えるとするとどのようにすれば良いか．改善案の 1 つのコードを，その案を提案した理由とともに述べなさい．コードと説明を合わせて 50～100 字程度とする．

(5) 関数 findRank を実装する．ソートには上記の関数 quickSort を用いるとする．そのうえで findRank 全体の平均の時間計算量をできるだけ低くするためにはどのような探索アルゴリズムを用いるべきか．使うべき探索アルゴリズムの名称と仕組みを簡単に説明し，その時間計算量を $O(\ )$記法で述べなさい．合わせて 50～100 字程度とする．

(6) 関数 findRank の最悪の時間計算量を $O(N\log(N))$ とするためには，ソートと探索にそれぞれどのようなアルゴリズムを用いれば良いか．合わせて 50～100 字程度で述べなさい．また，その場合のソート，探索，順位探索関数 findRank 全体，それぞれの時間計算量を $O(\ )$記法で記すこと．ソートおよび探索には，これまで述べた以外のアルゴリズムを使っても構わない．

【 前 期 募 集 】

令 和 6 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	試験科目	アルゴリズムとデータ構造，並びにプログラミング

```
using namespace std;

void insertionSort(int a[], int N) { // a[]を昇順にソート
  for (int i = 1; i < N; i++) {
    int j = i;
    while ( j > 0 && [      A      ] ) {
      swap(a[j], a[j - 1]);     // この関数 swap の実行回数を計数
      j--;
    }
  }
}

int partition(int a[], int low, int high) {
  int pivot = a[high];
  int i = low - 1;
  for (int j = low; j <= high - 1; j++) {
    if (a[j] <= pivot) {
      i++;
      swap(a[i], a[j]);         // この関数 swap の実行回数を計数
    }
  }
  swap(a[i + 1], a[high]);
  return i + 1;
}

void quickSort1(int a[], int low, int high) {
  if (low < high) {
    int pi = partition(a, low, high);
    [      B      ]
  }
}

void quickSort(int a[], int N) { // a[]を昇順にソート
  quickSort1(a, 0, N-1);
}
```

【 前期募集 】

令和６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	試験科目	アルゴリズムとデータ構造，並びにプログラミング

問４　C++言語で記述されたソースファイル bst.cpp は，二分探索木におけるデータの探索およびデータの削除を行うものである．以下の設問に答えなさい．

(1) ソースファイル中の空欄 [A] ～ [C] に当てはまるコードを記述して，データの探索を行う処理を完成させなさい．

(2) 二分探索木の各頂点のデータが図 1 の構造であったときを考える．ここで，メンバ関数 find を使ってデータ「18」の探索を行なったとき，このデータにたどり着いたと判定されるまでに参照される頂点の数を答えなさい．

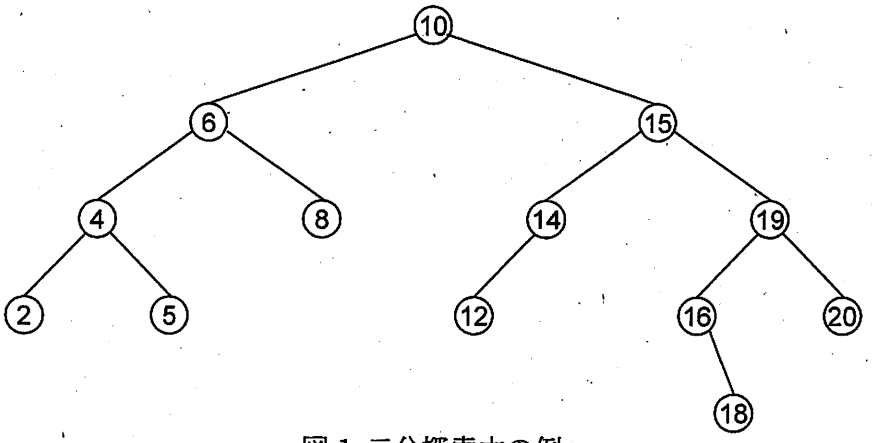

図 1 二分探索木の例

(3) 二分探索木の探索に必要な最悪の時間計算量は木の構造によって決まる．探索に必要な最悪の時間計算量が最大，最小となるのは，木がどのような構造のときか，それぞれ 30～50 字程度でそのときの木の構造を説明しなさい．

(4) 図 1 の構造をもつ二分探索木において，データ「15」を，その右部分木の最小値を移動する方法によって削除する．このとき，削除後の二分探索木の構造を図示しなさい．

(5) データの削除をその右部分木の最小値を移動する方法によって実装する．ソースファイル中の空欄 [D] ～ [H] に当てはまるコードを記述しなさい．

【 前 期 募 集 】

令和６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	試験科目	アルゴリズムとデータ構造，並びにプログラミング

bst.cpp

```
using namespace std;

template <class T> class BinNode {  // 二分木の頂点のクラス
private:
    T data;
    BinNode<T> *left, *right;
public:
    BinNode(T d, BinNode<T> *l = NULL, BinNode<T> *r = NULL) {
        data = d; left = l; right = r;
    }
    friend class BinarySearchTree;
};

class BinarySearchTree {
    BinNode <int>* root;  // 根頂点を指すポインタ
public:
    BinarySearchTree( ) { root = NULL; };  // 初期状態を定義
    ~BinarySearchTree( ) { makeEmpty( root ); };  // デストラクタ
    bool insert( int key ) { return insert( key, root ); };
                                                    // 根頂点を始点として追加
    bool find( int key ) const { return find( key, root ); };
                                                    // 根頂点を始点として探索
    bool remove( int key ) { return remove( key, root ); };
private:
    bool insert( int key, BinNode<int>* & tree);
    void makeEmpty( BinNode<int>* tree );
    bool find( int key, BinNode<int>* tree) const;
    bool remove( int key, BinNode<int>* & tree );
    int removeMin( BinNode<int>* & tree );
};

void BinarySearchTree::makeEmpty( BinNode<int>* tree ) {
    if ( tree != NULL ) {
        makeEmpty( tree->left );
        makeEmpty( tree->right );
        delete tree;
    }
}

// 次ページへ続く
```

令和６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	試験科目	アルゴリズムとデータ構造，並びにプログラミング

```
// 前ページからの続き

// データを追加する関数
bool BinarySearchTree::insert( int key, BinNode<int>* & tree ) {
    if (tree == NULL) {
        tree = new BinNode<int> ( key );  // 頂点の追加
        return true;
    }
    else if ( tree->data == key )  // key で与えられたデータが
        return false;                // すでに存在する場合は追加しない
    else if ( [ A ] )
        return insert( [ B ] );  // 左部分木を再帰的に探索
    else
        return insert( [ C ] );  // 右部分木を再帰的に探索
}

// データを探索する関数
bool BinarySearchTree::find( int key, BinNode<int>* tree ) const {
    if ( tree == NULL )   // key で与えられたデータは存在しない
        return false;
    else if ( tree->data == key )  // key で与えられたデータを発見
        return true;
    else if ( [ A ] )
        return find( [ B ] );  // 左部分木を再帰的に探索
    else
        return find( [ C ] );  // 右部分木を再帰的に探索
}

// 次ページへ続く
```

令和６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	試験科目	アルゴリズムとデータ構造，並びにプログラミング

```
// 前ページからの続き

// データを削除する関数
bool BinarySearchTree::remove( int key, BinNode<int>* & tree ) {
    if ( tree == NULL )  // key で与えられたデータは存在しない
        return false;
    else if ( tree->data == key ) {  // key で与えられたデータを発見
        if( tree->left == NULL || tree->right == NULL ) {
                                                    // 子が１つ以下の場合
            BinNode <int>* toBeRemoved = tree;
            if ( [ D ] != NULL )
                tree = [ D ];
            else
                tree = [ E ] ;
            delete toBeRemoved;
        }
        else  // 子が２つある場合
            tree->data = removeMin( [ F ] );
        return true;
    }
    else if ( [ A ] )
        return remove( [ B ] );  // 左部分木を再帰的に探索
    else
        return remove( [ C ] );  // 右部分木を再帰的に探索
}

//  最小のデータを持つ頂点を削除する関数
int BinarySearchTree::removeMin( BinNode<int>* & tree ) {
    assert( tree != NULL );  // 空の木に対して操作できないため
    if ( [ G ] != NULL )
        return removeMin( [ G ] );  // 部分木を再帰的探索
    else {  // 最小のデータを持つ頂点
        int min = tree->data;
        BinNode <int>* toBeRemoved = tree;
        tree = [ H ] ;
        delete toBeRemoved;
        return min;  // 最小値を返す
    }
}
```

令和６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

問５　C++言語で記述されたソースファイル train.cpp には，車両を表す Car クラス，および Car クラスオブジェクトの連結リストで列車を表す Train クラスが宣言・定義されている．このソースファイルについて，以下の設問に答えなさい．

(1)　main 関数中の(A)の行を実行した際の標準出力結果を答えなさい．

(2)　Train クラスのメンバ関数 func3_I と同じ機能を持つメンバ関数 func3_R を，関数の再帰呼び出しを用いて実装する．このとき，空欄 [B] に当てはまるコードを答えなさい．複数行のコードを解答してもよい．ただし，空欄 [B] 内で反復処理を用いてはいけない．

(3)　Train クラスのデストラクタ ~Train 内で何も行わない場合にどのような問題が発生するか，100～150 字程度で答えなさい．

(4)　設問 (3)で答えた問題を解消するための，空欄 [C] に当てはまるコードを答えなさい．複数行のコードを解答してもよい．

(5)　Car クラスオブジェクトの連結リストではなく，Car クラスオブジェクトの配列で列車を表現することも可能である．１つの列車が多数（例：100 万台）の車両から成る場合，列車の途中に車両を新たに追加したり，途中の車両を削除したりすることを考える．このとき，連結リストと配列のどちらの利用が適切であるか，理由とともに 150～200 字程度で答えなさい．理由には，連結リストと配列の間での時間計算量に関する比較を含めること．

令和６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. 11

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

train.cpp

```
#include <iostream>
using namespace std;

class Car{
  friend class Train;
private:
  double length;
  Car * preceding;
public:
  Car( double l, Car * p ){ length = l; preceding = p; }
};

class Train{
private:
  Car * tail;
public:
  Train(){ tail = NULL; };
  ~Train();
  void func1( double length );
  void func2();
  double func3_I( double speed );
  double func3_R( double speed ){ return func3_R( speed, tail ); };
  double func3_R( double speed, Car * c );
};

void Train::func1( double length ){
  tail = new Car( length, tail );
}

void Train::func2(){
  for( Car * p = tail; p != NULL; p = p->preceding ){
    cout << p->length << " ";
  }
  cout << endl;
}


// 次ページへ続く
```

令和６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. 12

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

```
// 前ページからの続き

double Train::func3_I( double speed ){
  double total_length = 0.0;
  for( Car * p = tail; p != NULL; p = p->preceding ){
    total_length += p->length;
  }
  return total_length / speed;
}

double Train::func3_R( double speed, Car * c ){

    B

}

Train::~Train(){

    C

}

int main(){
  Train t;
  t.func1( 30.0 );
  t.func1( 10.0 );
  t.func1( 25.0 );
  t.func1( 15.0 );
  t.func1( 20.0 );

  t.func2();    // ----(A)----

  cout << t.func3_I( 15.0 ) << endl;
  cout << t.func3_R( 15.0 ) << endl;
  return 0;
}
```