令和１年度後期　データ構造とアルゴリズム　期末テスト

※　各問題中のアルゴリズムを表すプログラムは，変数の宣言が省略されているなど，完全なものではありませんが，適宜，常識的な解釈をしてください．疑問があれば，挙手をして質問してください．
※　時間計算量をオーダ記法で表せという問題では，入力サイズ n を無限大に近づけた場合の漸近的な時間計算量を表せということだと考えてください．

問題１　入力サイズが n の問題を解くアルゴリズムの正確な時間計算量が次のような場合，それぞれの漸近的な時間計算量をオーダ記法で表せ．

（１）3　（２）$2n$　（３）$4n^2+3n$
（４）$5n^3-n-2$　（５）$2\sqrt{n}+3n-1$　（６）$3\log_{10}n+2n+1$
（７）$3\sqrt{n}+4\log_{10}n-2$　（８）$2n\log_{10}n+5n+1$　（９）$3n\log_{10}n+2n^2-4$
（10）$n^{100}+2^n$　（11）$2^n+n!$　（12）$n!+n^{100}$

問題２　次のプログラムはデータを**“連結リスト”**に格納する．連結リストのレコードは構造体 rec で表され，rec 中の d はデータ，p は次のレコードを指すポインタである．連結リストの先頭のレコードを指すポインタ head は「何もない」ことを表す値 NULL で初期化している．関数 add は連結リストの先頭へのデータ x の追加，関数 del は先頭からのデータの削除，関数 disp は連結リスト中の全データの表示を行う．関数 add では関数 malloc により新たなレコードを生成している．

```
// レコードの構造体
typedef struct rc {
    int d;
    struct rc *p;
} rec;
// 先頭レコードを指すポインタ
rec *head = NULL;

void add( int x )
{
    rec *r;     // 新たなレコードの生成
    r = (rec *)malloc(sizeof(rec));
    r->d = x;           // 追加データの格納
    r->p = head;        // (a)
    head = r;           // (b)
}

void del( void )
{
    rec *r;
    if( head != NULL ) {
        r = head->p;    // (c)
        free( head );   // 先頭レコードの削除
        head = r;       // (d)
    }
}
```

```
void disp( void )
{
    rec *r;
    r = head;
    while( r != NULL ) {
        printf( "%d ", r->d );
        r = r->p;
    }
    printf( "¥n" );
}

void main( void )
{
    add( 3 );  add( 7 );  add( 1 );
    disp();     // (★1)
    del();
    disp();     // (★2)
    add( 8 );  add( 4 );
    disp();     // (★3)
    del();  del();  del();
    disp();     // (★4)
}
```

（１）(a)と(b)の結果として，連結リストに対して新たなレコードがどうされるかを書け．

（２）(c)と(d)で何が行われているのかを書け．r に何が入るのかがわかるように書くこと．

（３）(★1)から(★4)のそれぞれで表示される全データを書け．表示される順序で書くこと．

（４）連結リストの末尾のレコードのポインタ p には何が入るかを書け．

問題３　整数 n は 2 のべき乗の数とする．このとき，定数 a に対して a^n を求める次の３つのアルゴリズムを考える．下の問いに答えよ．

```
/* アルゴリズムＡ */
powA( n ) {
    x = 1;
    for( i = 0; i < n; i++ ) {
        x = x * a;
    }
    return( x );
}
```

```
/* アルゴリズムＢ */
powB( n ) {
    if( n == 1 ) { return( a ); }
    else {
        return( powB( n/2 ) * powB( n/2 ) );
    }
}
```

```
/* アルゴリズムＣ */
powC( n ) {
    if( n == 1 ) { return( a ); }
    else {
        p = powC( n/2 );
        return( p * p );
    }
}
```

（1）アルゴリズム Ｂ と Ｃ のように関数が内部で自分自身を呼び出すようなアルゴリズムは何と呼ばれるか，名称をかけ．

（2）それぞれのアルゴリズムの時間計算量をオーダ記法で書け．

問題４　次のプログラムは**“ハッシュ法”**を実現するものであり，関数 hash_input はハッシュ関数 hash によるデータの格納，関数hash_searchはデータの探索を行う．各関数の引数については，n ≦ mとし，与えられた n 個のデータを D[0], ..., D[n-1]，ハッシュ関数によりデータを格納する大きさ m の配列を H[0] ..., H[m-1]，探索する値を x とする．また，NO_VAL は配列 H 中にデータが格納されていないことを示す値であり，データとして格納される値とは異なるものとする．下の問いに答えよ．

```
void hash_input(
    int n, int m, int D[], int H[] )
{
    // 配列 H の初期化
    for( i = 0; i < m; i++ ) {
        H[i] = NO_VAL;
    }
    // データの格納
    for( i = 0; i < n; i++ ) {
        k = hash( m, D[i] );
        while( H[k] != NO_VAL ) {
            k = ( k + 1 ) % m;    // (A)
        }
        H[k] = D[i];
    }
}
```

```
int hash_search(
    int m, int H[], int x )
{
    k = hash( m, x );
    while( H[k] != NO_VAL ) {
        if( H[k] == x ) {
            return( k );    // (B)
        }
        k = ( k + 1 ) % m;    // (A)
    }
    return( -1 );    // (C)
}

int hash( int m, int x )
{
    return( x % m );
}
```

（1）データの個数が n = 8 で，配列 D[0], …, D[7]に下記の値がその順序で入っているとする．また，配列 H の大きさを m = 12 とする．このとき，関数 hash_input の実行後に配列 H[0], …, H[11]に格納されている値を書け．値が格納されていない場所には × を書け．

(a) 1, 3, 7, 9, 15, 19, 23, 27

(b) 3, 9, 19, 27, 1, 7, 23, 15

(c) 3, 19, 1, 27, 8, 5, 15, 7

（2）プログラム中の２箇所の(A)で　% m　によってｍによる除算の余りを求める理由を書け．

（3）関数 hash_search の(B)で返り値とされる値 k は何かを書け．

（4）関数 hash_search の(C)で返り値が -1 とされるのはどのような場合かを書け．

問題５　次の関数selectionsortは**“選択ソート”**，関数insertionsortは**“挿入ソート”**によって，配列 D[0], ..., D[n-1]中の n 個のデータを昇順に並べ変える．関数 swap は２つの引数のデータを交換するものである．下の問いに答えよ．

```
void selectionsort( int n, double D[] )    // 選択ソート
{
    for( i = n-1; i > 0; i-- ) {
        max = D[0];
        max_index = 0;
        for( j = 1; j <= i; j++ ) {
            if( D[j] >= max ) {    // ← ①
                max = D[j];
                (a)
            }
        }
        swap( &(D[max_index]), &( (b) ) );
    }
}

void insertionsort( int n, double D[] )    // 挿入ソート
{
    for( i = 1; i < n; i++ ) {
        x = D[i];
        (c)
        while( ( j > 0 ) && ( D[j-1] > x ) ) {    // ← ②
            (d)
            j = j - 1;
        }
        D[j] = x;
    }
}
```

（１）それぞれの関数の空欄を埋めよ．

（２）データの個数が n = 5 で，ソート前の初期状態として配列 D[0], …, D[4]に下の(A)から(C)の値がその順序で入っていたとする．(A)から(C)のそれぞれの場合について，それぞれの関数が変数 i による外側の for ループの繰り返しで i の値を変えながら実行されるとき，変数 i の各値での実行が終了した直後の配列 D 中のデータを書け．また，変数 i の各値において，選択ソートでは①中の比較 D[j] >= max，挿入ソートでは②中の比較 D[j-1] > x が実行される回数を書け．

(A) 9, 1, 5, 3, 7　　(B) 1, 3, 5, 7, 9　　(C) 9, 7, 5, 3, 1

（３）それぞれの関数について，最良時間計算量，最悪時間計算量，平均時間計算量をオーダ記法で表せ．

問題６　次のプログラムは，quicksort(D, 0, n-1)を実行することで，配列 D[0], ..., D[n-1]中の n 個のデータを**“クイックソート”**によって昇順に並べ変える．なお，プログラム中の変数の宣言は省略している．下の問いに答えよ．

```
void quicksort(
    double D[],         // データ D[left], ..., D[right]
    int left,           // ソートの対象とする配列 D の左端の位置
    int right           // ソートの対象とする配列 D の右端の位置
)
{
    if( left < right ) {
        pivot_index = partition( D, left, right );        // (a)
        quicksort( D, left, pivot_index-1 );              // (b)
        quicksort( D, pivot_index+1, right );             // (c)
    }
}
```

（1）データの個数が n = 10 で，配列 D[0], …, D[9]に

35, 21, 4, 49, 55, 19, 12, 32, 24, 42

がこの順序で入っている状態で，関数 quicksort が left = 0, right = 9 として呼び出されたとする．基準値を D[0] = 35 とした場合，(a)で実行された関数 partition が戻り値（返り値）として変数 pivot_index に渡す値を答えよ．

（2）（1）の後，(b)のために関数 quicksort に渡される D[left], …, D[pivot_index-1]に入っているデータをすべて答えよ．順序は問わない．

（3）（2）の後，(c)のために関数 quicksort に渡される D[pivot_index+1], …, D[right]に入っているデータをすべて答えよ．順序は問わない．

（4）データが n 個のときの上記のプログラムの時間計算量 $T(n)$ は次式であらわされる．

$$T(n) = cn + T(n_l) + T(n_r)$$

右辺の c は定数であり，cn は(a)，$T(n_l)$ は(b)，$T(n_r)$ は(c)の時間計算量である．

（4－1）n_l と n_r が何をあらわすかを書け．また，n，n_l，n_r の関係を書け．

（4－2）D[left] から D[right] の中からどのような値を基準値として選ぶ場合に $T(n)$ が**最悪時間計算量**になるかを書け．また，このときの n_l，n_r の値を書け．さらに，その最悪時間計算量を n に関するオーダ記法で書け．

（4－3）D[left] から D[right] の中からどのような値を基準値として選ぶ場合に $T(n)$ が**最良時間計算量**になるかを書け．また，このときの n_l，n_r の値を書け（近似値でよい）．さらに，その最良時間計算量を n に関するオーダ記法で書け．

問題７ 次のプログラムは，mergesort(D, 0, n-1)を実行することで，配列 D[0], ..., D[n-1]中の n 個のデータを**“マージソート”**によって昇順に並べ変える．なお，プログラム中の変数の宣言は省略している．下の問いに答えよ．

```
void mergesort(
    double D[],        // データ D[left], ..., D[right]
    int left,          // ソートの対象とする配列 D の左端の位置
    int right          // ソートの対象とする配列 D の右端の位置
)
{
    mid = ( left + right ) / 2;                                    // (★1)
    if( left < mid ) { mergesort( D, left, mid ); }                // (★2)
    if( mid+1 < right ) { mergesort( D, mid+1, right ); }          // (★3)
    merge( D, left, mid, right );
}
void merge(
    double  D[],    // データ D[left], ..., D[right]
    int     left,   // ソートの対象とする配列 D の左端の位置
    int     mid,    // ソートの対象とする配列 D の中央の位置
    int     right   // ソートの対象とする配列 D の右端の位置
)
{
    x = left;    y = mid + 1;
    for( i = 0; i <= right-left; i++ ) {
        if( x == mid+1 ) { [ (a) ]; [ (b) ]; }
        else if( y == right+1 ) { [ (c) ]; [ (d) ]; }
        else if( D[x] <= D[y] ) { [ (e) ]; [ (f) ]; }
        else { [ (g) ]; [ (h) ]; }
    }
    for( i = 0; i <= right - left; i++ ) { D[left+i] = M[i]; }
}
```

（１）空欄(a)～(h)を埋めてプログラムを完成させよ．

（２）データの個数が n = 10 で，配列 D[0], …, D[9]に

35, 21, 4, 49, 55, 19, 12, 32, 24, 42

がこの順序で入っている状態で，関数 mergesort が left = 0, right = 9 で呼び出されたとする．

（２－１）(★1)で mid に代入される値を答えよ．

（２－２）(★2)の関数 mergesort の実行後の D[left], …, D[mid]中の値をその順序で答えよ．

（２－３）(★3)の関数 mergesort の実行後の D[mid+1], …, D[right]中の値をその順序で答えよ．

令和１年度後期　データ構造とアルゴリズム　期末テスト　解答用紙

平成　　年度入学　学籍番号＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿　氏名＿＿＿＿＿＿＿＿＿＿

問題１

（1）＿＿＿＿＿＿（2）＿＿＿＿＿＿（3）＿＿＿＿＿＿

（4）＿＿＿＿＿＿（5）＿＿＿＿＿＿（6）＿＿＿＿＿＿

（7）＿＿＿＿＿＿（8）＿＿＿＿＿＿（9）＿＿＿＿＿＿

(10)＿＿＿＿＿＿(11)＿＿＿＿＿＿(12)＿＿＿＿＿＿

問題２（1）

（2）

（3）

(★1)＿＿＿＿＿＿(★2)＿＿＿＿＿＿

(★3)＿＿＿＿＿＿(★4)＿＿＿＿＿＿

（4）

問題３（1）

（2）

A＿＿＿＿＿＿B＿＿＿＿＿＿

C＿＿＿＿＿＿

問題４（1）

H	[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]	[10]	[11]
(a)												
(b)												
(c)												

（2）

問題４（3）

（4）

問題５（1）

(a)＿＿＿＿＿＿(b)＿＿＿＿＿＿

(c)＿＿＿＿＿＿(d)＿＿＿＿＿＿

（2）

選択ソート (A)

i	D[0]	D[1]	D[2]	D[3]	D[4]	比較回数
初期状態	9	1	5	3	7	
4						
3						
2						
1						

選択ソート (B)

i	D[0]	D[1]	D[2]	D[3]	D[4]	比較回数
初期状態	1	3	5	7	9	
4						
3						
2						
1						

選択ソート (C)

i	D[0]	D[1]	D[2]	D[3]	D[4]	比較回数
初期状態	9	7	5	3	1	
4						
3						
2						
1						

挿入ソート (A)

i	D[0]	D[1]	D[2]	D[3]	D[4]	比較回数
初期状態	9	1	5	3	7	
1						
2						
3						
4						

問題５（２）

挿入ソート (B)

i	D[0]	D[1]	D[2]	D[3]	D[4]	比較回数
初期状態	1	3	5	7	9	
1						
2						
3						
4						

挿入ソート (C)

i	D[0]	D[1]	D[2]	D[3]	D[4]	比較回数
初期状態	9	7	5	3	1	
1						
2						
3						
4						

（３）

	最良	最悪	平均
選択ソート			
挿入ソート			

問題６（１）

（２）

（３）

（４）（４－１）

問題６（４－２）

（４－３）

問題７（１）

(a) (b)

(c) (d)

(e) (f)

(g) (h)

（２）（２－１）

（２－２）

（２－３）