データ構造とプログラミング技法　　学科＿＿＿＿学籍番号＿＿＿＿＿＿氏名＿＿＿＿＿＿＿＿＿＿＿＿

1. 以下の文章の空欄を埋めなさい．(各 1 点：合計 25 点)

- 関係 R が［　　律］（xRx），［　　律］（xRyかつyRz ならば xRz），反対称律（［　　ならば　　］）の３つを満足するとき関係 R を［　　関係］と呼ぶ．これらの条件に加え，任意の x , y について（［　　または　　］）が常に成り立つとき，関係 R を全順序関係と呼ぶ．
- 全順序関係は［　　構造］，［　　関係］は木構造や，［　　構造］における要素間の関係である．
- 物理構造としては，一連のデータを連続するメモリ番地に記憶する［　　］配置，メモリ上に分散したデータをポインタによって接続する［　　］配置の 2 種類がある．
- ヒープソートでは，キー列のデータ構造を，論理構造としては［　　］，物理構造としては［　　］配置と見なしている．
- ハッシュ法における衝突処理は，開番地法と［　　法］に分類される．開番地法の一つである［　　走査法］では，表サイズとハッシュ増分が互いに素である場合，探索周期は［　　］になる．
- キー列から逐次的に［　　探索木］を作成する際に，アンバランスな木構造になることがある．この問題を回避するために考案されたのが［　　木］である．この手法は木構造を作成している途中で発生する，平衡係数が［　，　］の範囲を超えた節のうち根から最も［　いものについて，［　　回転あるいは　　回転］の操作を行って再平衡化を行うというものである．
- k 次のＢ木では，根以外のノードに蓄積できるキーの個数は［　　個以上　　個以下］であり，根のノードでは［　　個以上　　個以下］と決まっている。
- BM 法は text 中に存在する部分文字列 pat を効率良く探し出すアルゴリズムである。同様のアルゴリズムである［　　法］は，［　　］内部の文字の並びを調べて照合再開位置を求めているが，BM 法では，これに加えて照合に失敗した［　　］側の文字種も利用する。

2. 下記はグラフ探索を行うプログラムである．空白部分を埋めなさい．（各 1 点：合計 5 点）

```
graph-search(vertex v)
    { push(S,v);
      while (!empty(S)) {［　　　　　］;
                         if (v.visit == ‘［　　］’) {v.visit= ‘［　　］’;
                                                     for (i=0; i<N;i++)
                                                       if ((adj[w[i]][v]==1)&&(w[i].visit== ‘［　　］’)) ［　　　　　］;
                                                     }
                        }
    }
```

但し，

- N は頂点数．w[i]は i 番目の頂点．adj[w][v]は隣接行列であり，頂点 w,v 間にエッジがあれば 1，なければ 0 の値がセットされている．
- 線形構造 S に対しては，vertex 型のデータ v を格納する push(S,v)と，v を取り出す pop(S,v)の操作のみが許されている．
- vertex 型のデータ v の visit 欄の初期値は’NO’であるとする．

3. データの追加を a[++n]=K で行う場合，下記の探索アルゴリズムはどんな場合に使うと有利であるか？下の１）～４）から適切なものを２つ選び○で囲みなさい。（両方正解で５点）

```
void func1(int K, int n)
{       int i;
        a[0]=K;
        for (i=n;a[i]!=K; i--) ;
        return i;
}
```

１）キーの追加よりも探索の要求が多い場合，２）キーの探索よりも追加の要求が多い場合，３）新しく追加されたキーに対する探索頻度が高い場合。４）過去に追加されたキーに対する探索頻度が高い場合。

4. ３．のプログラムをより効率よくするために，下記のように書きなおした。空白部分を埋めなさい。(6 点)

```
void func2(int K, int n)
{     int i;
        a[0]=K;
        for (i=n;a[i]!=K; i--) ;
       if ( i !=    && i !=    && a[i] ==    ) swap(&a[i],&a[++i]);
                                                    // swap(&a,&b) は a と b の値を入れ替える関数
       return i;
}
```

5. 年功序列を基本とする給与体系を採用している会社の社員について，氏名，年齢，給与の 3 つの欄から成る名簿データがある．全社員に対するソーティングを 2 回行い，このデータを年齢の若い順に並べ，且つ，同じ年齢の場合は給与が少ない順に並ぶようにしたい．1, 2 回目のソーティングに用いる欄はそれぞれ何か．また，2 回目のソーティングで使用すべきアルゴリズムとして何が適当かを理由とともに説明しなさい．(4 点)

6. 2,22,10,3,15, 19,14,11,8,6 を順に与えたときのハッシュ表と成功探索時の平均探査回数を求めなさい．但し，表サイズは 11， $h_0(x) = x \mod 11, h_i(x) = (h_0(x) + 2 \times i) \mod 11$ とする．(表 5 点，平均探査回数 3 点：合計 8 点)

表										
探査回数										

平均探査回数=　　回

7. 21, 2, 28, 14, 17,5, 6 というデータを順に与えたときに出来上がる二分探索木と，平衡が崩れた際に再平衡化を行って作成した AVL 木を図示し，成功探索時の平均探査回数をそれぞれ求めなさい．(2 分探索木 2 点，AVL 木 4 点，平均探査回数は各 2 点：合計 10 点)

8. 下記の KMP 法のプログラムにおいて, pat="ABABBA"の場合, compnext(pat)で計算している配列 next は, どのようになるか答えなさい．また，テキスト BABABBBABABBA から KMP 法で pat を見つける際の pat の配置と照合の様子を下図に書き込み，文字比較回数を答えなさい．（4 点+6 点：合計 10 点）

```
intkmp(char *text, char *pat)
{   inti=1, j=1;
compnext(pat);
while (j<=n) {
          while(i>0 && pat[i]!=text[j]) i=next[i];
          if (i!=m ) {i++; j++;}
          else return j-m+1;
    }
    return -1;
}
```

	A	B	A	B	B	A
next⇒						

BABABBBABABBA（テキスト）

文字比較回数 ____回

9. 単純挿入法や線形探索などで用いられていた門番（番人, sentinel）は，アルゴリズムの上では用いても用いなくてもよい．それにもかかわらず，この技法が良く用いられる理由を説明しなさい．(5 点)

10. 符号付き 8bit 整数のキー列，－１２７，－５０，５０，６，９をトライに格納した場合と，Patricia 木に格納した場合の木構造を示しなさい．但し，負の数の表現には２の補数を用いるものとする．(10 点)

11. BM 法が日本語を対象とした文字検索であまり用いられない理由を説明しなさい。(5 点)

12. 下記のアルゴリズムの名称と目的を説明しなさい．また，空白部分を埋めなさい．（2+2+3 =7 点）

```
void func1(int p)
{           inti,u,a;
            doubletmp;
            add(p,S);
            for (i=0;i<N; i++) m[i]=L[p][i];
            while (remain()) {
                        u = select_minimum();
                        add(u,S);
                        for (a=0; a<N; a++)
                                    if (member(u,a)) {if (tmp= [          ] <m[a]) m[a]=tmp;}
 }
}
```

アルゴリズムの名称　＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿

目的　＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿＿を求めるアルゴリズム．

計算スペース---

データ構造とプログラミング技法　　学科＿＿＿＿学籍番号＿＿＿＿＿＿氏名＿＿＿＿＿＿＿＿＿＿＿＿

1. 以下の文章の空欄を埋めなさい．(各 1 点：合計 25 点)

- 関係 R が[反射律]（xRx），[推移律]（xRyかつyRzならばxRz），反対称律（[xRyかつyRxならば$x = y$]）の３つを満足するとき関係 R を[　半順序 関係]と呼ぶ．これらの条件に加え，任意の x, y について（[xRy または yRx]）が常に成り立つとき，関係 R を全順序関係と呼ぶ．
- 全順序関係は[　線形　構造]，[　半順序　関係]は木構造や，[　束　構造]における要素間の関係である．
- 物理構造としては，一連のデータを連続するメモリ番地に記憶する[　順　]配置，メモリ上に分散したデータをポインタによって接続する[　リンク　]配置の 2 種類がある．
- ヒープソートでは，キー列のデータ構造を，論理構造としては[　完全二分木　]，物理構造としては[　順　]配置と見なしている．
- ハッシュ法における衝突処理は，開番地法と[　連鎖　法]に分類される．開番地法の一つである[　線形　走査法]では，表サイズとハッシュ増分が互いに素である場合，探索周期は[　全表的　]になる．
- キー列から逐次的に[　二分探索木]を作成する際に，アンバランスな木構造になることがある．この問題を回避するために考案されたのが[　AVL 木]である．この手法は木構造を作成している途中で発生する，平衡係数が[-1,1]の範囲を超えた節のうち根から最も[遠い]ものについて，[　単回転あるいは 複回転]の操作を行って再平衡化を行うというものである．
- k 次のＢ木では，根以外のノードに蓄積できるキーの個数は[　k 個以上　2k 個以下]であり，根のノードでは[　1 個以上　2k 個以下]と決まっている。
- BM 法は text 中に存在する部分文字列 pat を効率良く探し出すアルゴリズムである。同様のアルゴリズムである[　kmp 法]は，[　pat]内部の文字の並びを調べて照合再開位置を求めているが，BM 法では，これに加えて照合に失敗した[　text]側の文字種も利用する。

2. 下記はグラフ探索を行うプログラムである．空白部分を埋めなさい．（各 1 点：合計 5 点）

```
graph-search(vertex v)
    { push(S,v);
      while (!empty(S)) {pop(S,v);
                         if (v.visit == 'NO') {v.visit= 'YES';
                                               for (i=0; i<N;i++)
                                                    if ((adj[w[i]][v]==1)&&(w[i].visit== 'NO')) push(S,w[i]);
                                              }
                        }
    }
```

但し，

- N は頂点数．w[i]は i 番目の頂点．adj[w][v]は隣接行列であり，頂点 w,v 間にエッジがあれば 1，なければ 0 の値がセットされている．
- 線形構造 S に対しては, vertex 型のデータ v を格納する push(S,v)と,v を取り出す pop(S,v)の操作のみが許されている．
- vertex 型のデータ v の visit 欄の初期値は'NO'であるとする．

3. データの追加を a[++n]=K で行う場合，下記の探索アルゴリズムはどんな場合に使うと有利であるか？下の１）～４）から適切なものを２つ選び○で囲みなさい。（両方正解で５点）

```
void func1(int K, int n)
{       int i;
        a[0]=K;
        for (i=n;a[i]!=K; i--) ;
        return i;
}
```

１）キーの追加よりも探索の要求が多い場合，（２）キーの探索よりも追加の要求が多い場合，（３）新しく追加されたキーに対する探索頻度が高い場合。４）過去に追加されたキーに対する探索頻度が高い場合。

4. ３．のプログラムをより効率よくするために，下記のように書きなおした。空白部分を埋めなさい。(6 点)

```
void func2(int K, int n)
{     int i;
        a[0]=K;
        for (i=n;a[i]!=K; i--) ;
        if ( i != [ n ] && i != [ 0 ] && a[i] == [ K ] ) swap(&a[i],&a[++i]);
                                                    // swap(&a,&b) は a と b の値を入れ替える関数
        return i;
}
```

5. 年功序列を基本とする給与体系を採用している会社の社員について，氏名，年齢，給与の 3 つの欄から成る名簿データがある．全社員に対するソーティングを 2 回行い，このデータを年齢の若い順に並べ，且つ，同じ年齢の場合は給与が少ない順に並ぶようにしたい．1, 2 回目のソーティングに用いる欄はそれぞれ何か．また，2 回目のソーティングで使用すべきアルゴリズムとして何が適当かを理由とともに説明しなさい．(4 点)

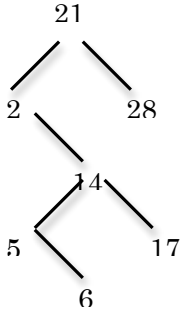
まず 1 回目は，給与についてソーティングを行い，2 回目は年齢についてソーティングを行う.この場合,2 回目のソートは安定なソートでなければならない．安定なソートには単純挿入法，単純交換法，基数ソートなどがあるが，この場合には年功序列であるので，1 回目のソートで概ソート列が得られているため，単純交換法，単純挿入法などが適している．

6. 2,22,10,3,15, 19,14,11,8,6 を順に与えたときのハッシュ表と成功探索時の平均探査回数を求めなさい．但し，表サイズは 11，$h_0(x) = x \mod 11, h_i(x) = (h_0(x) + 2 \times i) \mod 11$ とする．(表 5 点，平均探査回数 3 点：合計 8 点)

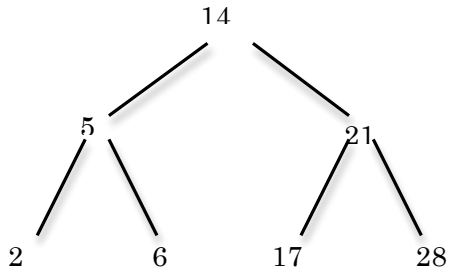
表	22	8	2	3	15	14	11	6	19		10
探査回数	1	3	1	1	1	2	4	7	1		1

平均探査回数= 2.2 回

7. 21, 2, 28, 14, 17,5, 6 というデータを順に与えたときに出来上がる二分探索木と，平衡が崩れた際に再平衡化を行って作成した AVL 木を図示し，成功探索時の平均探査回数をそれぞれ求めなさい．(2 分探索木 2 点，AVL 木 4 点，平均探査回数は各 2 点：合計 10 点)



平均探査回数：3 回



平均探査回数：2.2 回

8. 下記の KMP 法のプログラムにおいて, pat="ABABBA"の場合, compnext(pat)で計算している配列 next は, どのようになるか答えなさい．また，テキスト BABABBBABABBA から KMP 法で pat を見つける際の pat の配置と照合の様子を下図に書き込み，文字比較回数を答えなさい．（4 点+6 点：合計 10 点）

```
int kmp(char *text, char *pat)
{   int i=1, j=1;
compnext(pat);
while (j<=n) {
        while(i>0 && pat[i]!=text[j]) i=next[i];
        if (i!=m ) {i++; j++;}
        else return j-m+1;
    }
    return -1;
}
```

next⇒

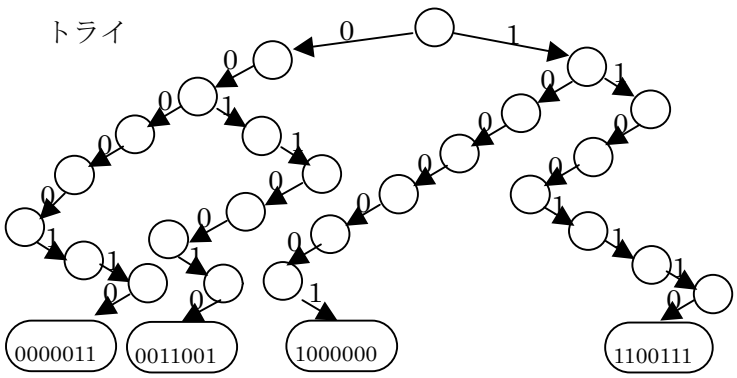
A	B	A	B	B	A
0	1	0	1	3	0

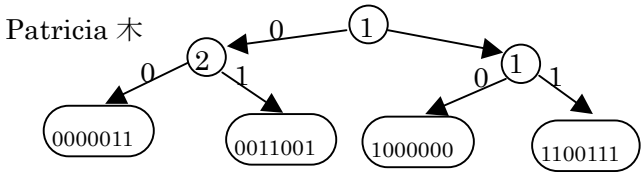
```
BABABBBABABBA（テキスト）
A
 ABABBA
        ABABBA


文字比較回数    13   回
```

9. 単純挿入法や線形探索などで用いられていた門番（番人, sentinel）は，アルゴリズムの上では用いても用いなくてもよい．それにもかかわらず，この技法が良く用いられる理由を説明しなさい．(5 点)

門番を用いると，ループ内部の条件分岐命令が一つ少なくなる．この命令が減ることにより，1 命令が減る以上に速度が向上する．これは，CPU 内部の命令パイプラインの実行が条件分岐命令の実行によって滞るためである．このパイプラインの乱れは，条件が確定しない限り次の命令が格納されている番地が確定できないという条件分岐命令の性質に依るものであり，この命令を減らせば確実に実行速度が向上する．最近の CPU には, 条件部の計算が終了していない場合にも, 実行される可能性のある命令を全て実行しておいて，条件部の計算が済んでから，実行されない命令の実行結果を捨て去る「投機的実行」機構が組み込まれたものもあるが，それでも条件分岐が重なると実行速度が低下するため，門番の使用は効果的である．

10. 符号付き 8bit 整数のキー列，－１２７，－５０，５０，６，９をトライに格納した場合と，Patricia 木に格納した場合の木構造を示しなさい．但し，負の数の表現には２の補数を用いるものとする．(10 点)





11. BM 法が日本語を対象とした文字検索であまり用いられない理由を説明しなさい。(5 点)

BM 法は，照合に失敗したときの pat 側の文字を利用して，ずらしの量と照合再開位置を求める KMP 法のアイデアに加えて，照合に失敗した text 側の文字が pat 側の末尾から見てどこに現れるかも調べてずらしの量と照合再開位置を求めている。このため，text や pat を構成している文字の種類分の配列を持たなければならず，その初期化も必要である。全角の日本語は，2 バイト文字で表現され 65536 個の文字種がある。このため，各文字を 1 バイトで表現できる半角英数字を対象とした場合と比べると，大きな配列を用意し，その初期化のための計算時間もかかる。このため，日本語を対象とした場合は，メモリ使用の制限が少なく，text が非常に長い場合以外では，KMP 法を利用したほうが良いためである。

12. 下記のアルゴリズムの名称と目的を説明しなさい．また，空白部分を埋めなさい．（2+2+2=6 点）

```
void func1(int p)
{       int i,u,a;
        double tmp;
        add(p,S);
        for (i=0;i<N; i++) m[i]=L[p][i];
        while (remain()) {
                u = select_minimum();
                add(u,S);
                for (a=0; a<N; a++)
                        if (member(u,a)) {if (tmp= m[u]+L[u][a]<m[a]) m[a]=tmp;}
 }
}
```

アルゴリズムの名称　Dijkstra のアルゴリズム

目的　（グラフ上の）頂点 p から各頂点までの距離（重みの和の最小値）を求める

計算スペース----------