令和６年度　豊橋技術科学大学第３年次入学者選抜学力検査問題

専　門　科　目　（３：情報・知能工学）

注　意　事　項

1　試験開始の合図まで，この問題冊子と解答用紙を開いてはいけません。

2　問題冊子の枚数は表紙，草稿用紙を含めて１０枚です。

3　問題冊子とは別に解答用紙が６枚あります。解答は用紙の裏面にまわってはいけません。

4　問題は３問あります。全問解答してください。

5　試験開始の合図の後すぐに，すべての解答用紙の所定の箇所に受験番号を記入してください。

6　解答は必ず各問題別の解答用紙の所定の欄に記入してください。

7　落丁，乱丁，印刷不鮮明の箇所などがあれば，ただちに申し出てください。

8　問題冊子の余白は草稿用として使用しても構いません。

9　試験終了時刻まで退出してはいけません。

10　問題冊子は持ち帰ってください。

（草　稿　用　紙）

［1］　実数tおよび実行列Aに対する指数関数$\exp(t)$および$\exp(A)$を次式で定義する。

$$\exp(t) = 1 + t + \frac{t^2}{2!} + \frac{t^3}{3!} + \cdots, \quad \exp(A) = I + A + \frac{A^2}{2!} + \frac{A^3}{3!} + \cdots$$

ここで，I＝単位行列，$A^2 = AA$, $A^3 = AAA$, …とする。このとき

$$\exp(tA) = I + tA + \frac{t^2}{2!}A^2 + \frac{t^3}{3!}A^3 + \cdots$$

が成り立つ。また，ある行列Uを用いて行列Aが$U^{-1}AU = D = \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix}$と対角化できるとき，

$$\exp(A) = \exp(UDU^{-1}) = U\exp(D)\,U^{-1}$$

が成り立つ。以下の各問いに答えよ。ただし，行列の要素はすべて実数とする。

(1)　自然数nと対角行列$D = \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix}$に対し，$D^n = \begin{pmatrix} \alpha^n & 0 \\ 0 & \beta^n \end{pmatrix}$となることを示せ。

(2)　対角行列$D = \begin{pmatrix} \alpha & 0 \\ 0 & \beta \end{pmatrix}$に対し，$\exp(D) = \begin{pmatrix} \exp(\alpha) & 0 \\ 0 & \exp(\beta) \end{pmatrix}$となることを示せ。

(3)　これらの指数関数を用いて，次の連立1次微分方程式を解くことを考える。

$$\begin{cases} \dfrac{dx_1(t)}{dt} = 3x_1(t) + x_2(t) \\ \dfrac{dx_2(t)}{dt} = x_1(t) + 3x_2(t) \end{cases}$$

ただし初期条件は$x_1(0) = 1$, $x_2(0) = 3$とする。

ア．いま2つのベクトルをそれぞれ$\boldsymbol{x}(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix}$, $\dot{\boldsymbol{x}}(t) = \begin{pmatrix} \frac{dx_1(t)}{dt} \\ \frac{dx_2(t)}{dt} \end{pmatrix}$と定義すると，上記の連立1次微分方程式は，ある$2\times2$行列$A$を用いて$\dot{\boldsymbol{x}}(t) = A\boldsymbol{x}(t)$と書ける。この行列$A$を示せ。

イ．この行列Aの2つの固有値λ_1, λ_2 $(\lambda_1 \geq \lambda_2)$とそれらに対応する固有ベクトルを求めよ。ただし，固有ベクトルは単位ベクトルとし，第1要素が正となるように符号を決めること。

ウ．行列Aを対角化，つまり$D = U^{-1}AU = \begin{pmatrix} \lambda_1 & 0 \\ 0 & \lambda_2 \end{pmatrix}$とするための行列$U$を示せ。

エ．上記微分方程式の解は$\boldsymbol{x}(t) = \begin{pmatrix} x_1(t) \\ x_2(t) \end{pmatrix} = \exp(tA)\,\boldsymbol{x}(0)$で与えられる。ただし$\boldsymbol{x}(0)$は初期条件であり，$\boldsymbol{x}(0) = \begin{pmatrix} x_1(0) \\ x_2(0) \end{pmatrix} = \begin{pmatrix} 1 \\ 3 \end{pmatrix}$となる。この微分方程式の解を求めよ。

[2] ノード群が木状に連結されている構造を表したものをツリーと呼ぶ。図1にツリーの例を示す。図1では，円がノードであり，ノードの連結を矢印で表している。この矢印は親子関係を表し，矢印が指している先を子ノードと呼び，矢印の元を親ノードと呼ぶ。

リスト1は，図1で表現されたツリーを扱うプログラムである。リスト1の4行目から9行目にノードを表す構造体を定義し，リスト1の11行目にその構造体の配列を用意してある。リスト1の18行目から23行目のように配列に初期化処理を行い，リスト1の25行目から51行目においてツリーの作成を行っている。ノードを表す構造体の ID はグラフの円内の数字，parent_ID は親ノードの ID，child_ID[] は子ノードの ID の配列を表し，number_of_children が子ノードの数を表す。

ツリーを処理するプログラムについて以下の各問いに答えよ。

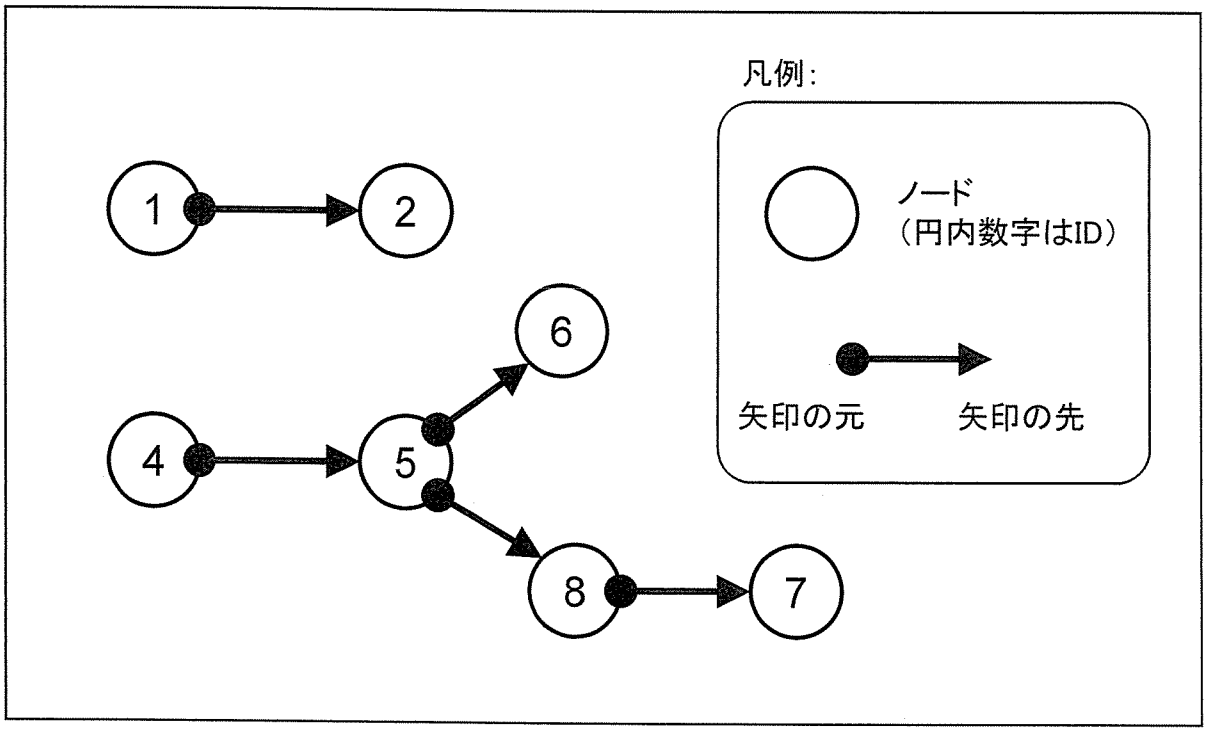


図1： ツリーの例

(1) リスト1の11行目で宣言した構造体の配列が図1に表したツリーとなるように，リスト1内の［ア］から［オ］に当てはまる数字を答えよ。

```
001:#include <stdio.h>
002:#define MAX_NODE_NUM 20
003:/* ノードを表す構造体定義 */
004:struct node {
005:  int ID;
006:  int parent_ID;
007:  int child_ID[MAX_NODE_NUM];
008:  int number_of_children;
009:};
010:/* ノードデータ変数 */
011:struct node node_list[MAX_NODE_NUM];
012:[          A          ]
013:[          B          ]
014:
015:int main(int argc, char **argv)
016:{
017:  /* 初期化処理 */
018:  int i;
019:  for (i = 0; i < MAX_NODE_NUM; i++) {
020:    node_list[i].ID = -1;
021:    node_list[i].parent_ID = -1;
022:    node_list[i].number_of_children = 0;
023:  }
024:  /* 構造初期化 */
025:  node_list[0].ID = 1;
026:  node_list[0].child_ID[0] = 2;
027:  node_list[0].number_of_children = 1;
028:
029:  node_list[1].ID = 2;
030:  node_list[1].parent_ID = [ ア ] ;
031:
032:  node_list[2].ID = 4;
033:  node_list[2].child_ID[0] = 5;
034:  node_list[2].number_of_children = [ イ ] ;
035:
036:  node_list[3].ID = 5;
037:  node_list[3].parent_ID = 4;
038:  node_list[3].child_ID[0] = [ ウ ] ;
039:  node_list[3].child_ID[1] = 8;
040:  node_list[3].number_of_children = [ エ ] ;
041:
042:  node_list[4].ID = 6;
043:  node_list[4].parent_ID = 5;
044:
045:  node_list[5].ID = 8;
046:  node_list[5].parent_ID = 5;
047:  node_list[5].child_ID[0] = 7;
048:  node_list[5].number_of_children = 1;
049:
050:  node_list[6].ID = [ オ ] ;
051:  node_list[6].parent_ID = 8;
052:
053:[          C          ]
054:
055:}
```

リスト1

(2) リスト2はリスト1の12行目 A で定義される。また，リスト1の53行目 C にて呼び出される。リスト2では，あるノードが属するツリーのルートノード（親がいないノード）を探索する関数が定義される。以下のアからオの各問いに答えよ。

```
001:int search_node_index(int ID)
002:{
003:  int i;
004:  for (i = 0; i < MAX_NODE_NUM; i++) {
005:    if (node_list[i].ID == ID) {
006:      return i;
007:    }
008:  }
009:  return -1;
010:}
011:int search_root_ID(struct node node)
012:{
013:  int index;
014:  struct node current_node;
015:  current_node = node;
016:  while (1) {
017:    if (current_node.parent_ID < 0) {
018:      return current_node.ID;
019:    }
020:    index = search_node_index(current_node.parent_ID);
021:    if (index < 0) { /* 条件分岐*/
022:      return  index;
023:    }
024:    [          D          ]
025:  }
026:}
```

リスト2

ア．リスト1の53行目 C において，次の関数実行を行ったときの戻り値を答えよ。

```
search_node_index(5);
```

イ．リスト2の21行から23行目の条件分岐及び処理の説明として最も適切でないものを以下の①から③より選べ。

① 負のindexではその後のプログラムが正しく動作しない可能性が高いため，関数 search_root_ID を終了する。
② search_node_index の引数に配列に存在しないノード ID を渡した場合に条件分岐し，リスト2の22行目の処理が行われる。
③ 目的の index を発見したので，値を戻し関数 search_root_ID を終了する。

ウ．リスト2の24行目 D に入れる記述として適切なものを，以下の ①から④より選べ。

```
① index++;
② current_node = node_list[index];
③ current_node = node_list[index++];
④ current_node = node_list[search_node_index(current_node.ID)];
```

エ．リスト1の53行目 C にて，以下の関数実行を行ったときの戻り値を答えよ。

```
search_root_ID( node_list[search_node_index(1)] );
```

オ．リスト1の53行目 C にて，以下の関数実行を行ったときの戻り値を答えよ。

```
search_root_ID( node_list[search_node_index(8)] );
```

(3) リスト3はリスト1の13行目 B に，リスト2に続いて定義される。リスト3では，あるノードのサブツリーの探索を行う関数が定義される。サブツリーとは，あるノードをルートノードとするツリーのことである。リスト3の1行目から2行目に大域変数の宣言があり，関数の結果を保持するために使われる。

```
001:int result_number_of_nodes = 0;
002:int result_node_ID_list[MAX_NODE_NUM];
003:void search_all_nodes(struct node node)
004:{
005:  int i;
006:  int ID;
007:  int index;
008:  result_node_ID_list[result_number_of_nodes] = node.ID;
009:  result_number_of_nodes++;
010:  if (node.number_of_children == 0) {
011:    printf("A: %d ", node.ID);
012:    return;
013:  }
014:  for (i = 0; i < node.number_of_children; i++) {
015:    ID = node.child_ID[i];
016:    index = search_node_index(ID);
017:    printf("B: %d ", ID);
018:    search_all_nodes(node_list[index]);
019:  }
020:  return;
021:}
```

リスト3

ア.リスト1の53行目 C にて，以下の関数実行が行われた場合において，以下の各問いに答えよ。

```
result_number_of_nodes = 0;
search_all_nodes( node_list[search_node_index(2)] );
```

(a) 関数実行が行われた場合の画面の表示を，以下の①から④より選べ。

① A: 1
② B: 1
③ A: 2
④ B: 2

(b) 関数実行終了後の result_number_of_nodes の値を答えよ。

(c) 関数実行終了後の result_node_ID_list[0] の値を答えよ。

イ．リスト1の53行目 C にて，以下の関数実行が行われた場合において，以下の各問いに答えよ。

```
result_number_of_nodes = 0;
search_all_nodes( node_list[search_node_index(4)] );
```

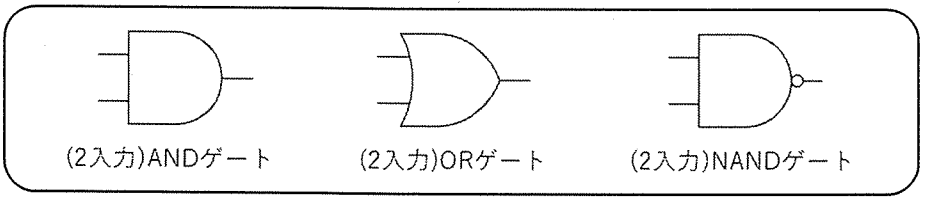
(a) 関数実行が行われた場合の画面の表示を，以下の①から④より選べ。

① B:2 A:1
② B:5 B:6 B:7 B:8 A:8
③ A:4 B:5 A:5 B:6 A:6 B:7 A:7 B:8 A:8
④ B:5 B:6 A:6 B:8 B:7 A:7

(b) 関数実行終了後の result_number_of_nodes の値を答えよ。

(c) 関数実行終了後の result_node_ID_list[2]の値と result_node_ID_list[3]の値をそれぞれ，解答欄(ⅰ)，(ⅱ)に記入せよ。

［３］以下の問いに答えよ。ただし，解答では，論理積には「・」，論理和には「＋」，変数Aの否定には「$\bar{A}$」を用い，真理値表，および，カルノー図では真の値は1，偽の値は0，ドントケアは×を用いて答えよ。最小積和形とは，複数個の論理積項が論理和で結ばれている論理式において，論理積と論理和の総数が最小になる形式である。ANDゲート，ORゲート，NANDゲートはそれぞれ下図のように表現される。



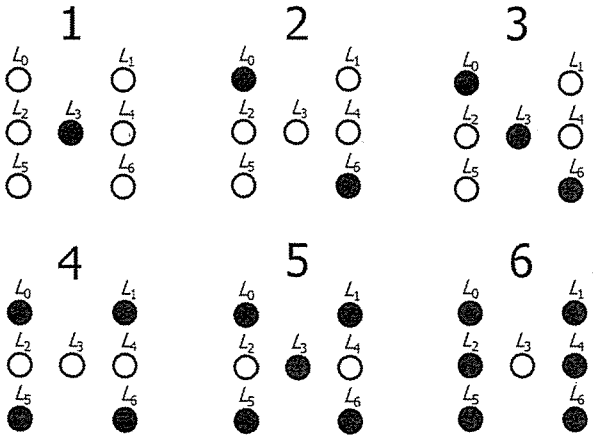
(1) 論理関数 $Y(A,B,C)=\bar{A}\cdot(B+C)+\bar{B}\cdot C$ について，以下の問いに答えよ。

ア．解答用紙の真理値表を埋めよ。

イ．論理関数Yを最小積和形で表せ。

ウ．論理関数Yを2入力NANDゲートのみで実現することを考える。イで求めた最小積和形の論理式を否定と論理積のみの式に書き換えよ。また、解答欄の回路図を完成させよ。

(2) L_0からL_6の７つのLEDを並べてサイコロの各面を作成する。サイコロは「1」から「6」の状態をとり，$C_2C_1C_0$の3bitで表されるものとする。「1」から「6」の各状態はそれぞれ順に000, 001, 011, 010, 110, 100で表されるものとする。サイコロを表すLEDは各状態において下図のように光るものとする。なお，LEDは点灯状態（●）を1とし，消灯状態（○）を0とする。



ア．解答用紙の真理値表を埋めよ。

イ．C_2，C_1，C_0を入力とし，各LED（L_0からL_6）の状態を出力とする最小積和形の論理式を，カルノー図を用いて求めよ。なお，解答欄にはカルノー図も示すこと。

ウ．イで求めた最小積和形の論理式を実現することを考える。解答欄の回路図にANDゲートおよびORゲートを用いた回路を追加し、各LED（L_0からL_6）の組合せ回路を完成させよ。なお，3入力ゲート素子を用いてもよい。