# 1121 NTOUCSE 程式設計 1C Final Exam

姓名：_____ 系級：_____ 學號：_____ 112/12/26 (二)

考試時間：**13:20 – 16:00**

Exam rules：
1. **Close book**, **close** everything including quizs, homeworks, assignments, reference materials, etc.
2. You can answer the questions in **English** or in **Chinese**, in this **problem sheet**, the **answer sheet**, or **both**.
3. You can use language features not taught in class if you feel necessary, but strictly limited in **C**.
4. **No** mobile phone, pad, computer or calculator is allowed. (Electronic) English dictionary is OK
4. No peeping around! No discussion! No exchange of any material; **raise your hand if you have any question about the exam problems**
5. If you turn in the paper earlier than the specified time, **leave the classroom immediately and quietly**
6. Against any of the above rules will be treated as cheating in the exam and handled by school regulations.
7. **Turn in BOTH the problem sheet with your name and id and answer sheet with your name and id**.

1. Please associate the compilation errors to the following 4 lines of program? [8] (the compilation error messages are given by Visual C/C++ compiler. The order is scrambled. Some program line will gives multiple error messages. Please give your answers with error number C####.) Please explain the reason for the error and try correct the errors?[12]

```
01. int n=3, array1[n];
02. int array2[3]={1,2,3,4};
03. int *array3="abcd";
04. array1 = array2;
```

error C2440: 'initializing ' : can not convert from 'const char [5]' to 'int *'
these two types are not related; requires reinterpret_cast, C-Style or function style conversion
error C2133: 'array1' : unknown size
error C2466: can not allocate array with constant size 0, constant must be an integer greater than 0
error C2106: '=' : left operand must have l-value
error C2440: '=' : can not convert from 'int [3]' to 'int []'
error C2057: requires a constant expression
error C2078: too many initializers

2. [5] In the following piece of program, how do you define an alias name *array5* for the integer array *array4* with pointer syntax in C such that these two names refer to the same array.

```
int i, array4[20];
_____ array5=array4;
for (i=0; i<20; i++)
    array5[i] = i;
```

3. (a) [5] Please use struct to define a CreditCard structure for storing the information of a physical credit card. The information includes 1. 16-digit card number, 2. Effective month/effective year, 3. Name of card holder (maximum 50 characters) 4. 3-digit card verification value/code (CVV/CVC). Example data is 1. 5412 3456 7890 1232, 2. 12/25 3. WANG, DAVID, 4. 911. The names and data types of each fields are specified as 1. cc_number character array, 2.1 expiration_month integer, 2.2 expiration_year integer, 3. user_name character array, 4. cvv character array

(b) [4] Please use the new type struct CreditCard to define a struct variable mycard and use the above example data to initialize the struct variable.

(c) [6] Assume that we have an struct CreditCard data array ccArray ready at hand with the number of elements in the integer variable nCards. Please complete the following program: Use dynamic memory allocation to allocate a new array ccArraySorted and copy the content from ccArray

```
#include _____________________
/* definition of struct CreditCard */
/* inside main() */
    _____________________ ccArraySorted=NULL;
    ccArraySorted = (_____________________) malloc(nCards*_____________________);
    if (_____________________) {
        printf("memory allocation failed\n");
        return 1;
    }
    for (int i=0; i<nCards; i++)
        _____________________
```

(d) [5] Continuing with part (c), please use the qsort() function in stdlib.h and the strcmp() function in string.h to sort the elements in array ccArraySorted in ascending order according to the credit card number

```
qsort(_____, _____, _____, _____________________________);
/* it cc_number is the first field of struct CreditCard, there is no need to define a separate
compare function */
```

(e) [6] Please use the bsearch() function in stdlib.h to complete the following piece of program to search the mycard defined in part (b) in the ccArraySorted struct array and print out the name of the card holder if the entry is found; print out the card number if the entry is not in the array

```
_____________ result;
result = (_____________) bsearch(_____, /*arguments 2~5 are the same as part (d) */);
if (_____________)
    printf("%s\n", _____________);
else
    printf("Card %s not found!\n", _____________);
```

(f) [4] How do you deallocate the memory when the contents of ccArraySorted is no more useful to the program? (the deallocation utility function will error if the augument pointer is NULL or the memory has been deallocated before, therefore it is a common practice to add some guarding condition check statement)

```
if (_____________) { _____(_____________); _____________; }
```

4. Using C grammar to answer the following

(a) [4] How do you define an 11-element integer array in line 02 such that the for loop in line 03~04 can use array[-5], array[-4], …, array[0], array[1],…, array[5] directly to access the contents in this array?

```
01 int i;
```

```
02 int _____;
03 for (i=-5; i<=5; i++)
04         array[i] = i;
```

(b) [4] How do you define in continuous memory a conceptually two dimensional array named array2 in line 02 to store 6*11integers, the index of first dimension is from 50 to 55 and the index of the second dimension is from -5 to 5, such that the loops in line 07~09 can use array2[50][-5], array2[50][-4], …, , array2[50][5], array2[51][-5], array2[51][-4], …, , array2[51][5], …, array2[55][-5], array2[55][-4], …, , array2[55][5] to access the contents of these 66 elements?

```
05 int i, j;
06 int _____;
07 for (i=50; i<=55; i++)
08         for (j=-5; j<=5; j++)
09                 array2[i][j] = i*11+j;
```

(c) [5] Follwing the definitions of array and array2 in part (a) and part (b), please write down the outputs of the following printf statements

```
printf("%d ", *(array-2)); _____
printf("%d ", array[-4]+5); _____
printf("%d ", **(array2+50)); _____
printf("%d ", *(*(array2+51)-5)); _____
printf("%d ", array2[55]-array2[50]); _____
```

5. [16] An intuitive implementation of depth-first search (DFS) is fixed layered multiple loops as shown by the following program. A recursive implementation let the number of layers be adjustable as required. The following program enumerates the 16 binary 4-digit numbers in ascending order as shown in the figure on the right side.

```
for (i[0]=0; i[0]<=1; i[0]++)
    for (i[1]=0; i[1]<=1; i[1]++)
        for (i[2]=0; i[2]<=1; i[2]++)
            for (i[3]=0; i[3]<=1; i[3]++) {
                for (j=0; j<4; j++)
                    printf("%d ", i[j]);
                printf("\n");
            }
```

```
0 0 0 0
0 0 0 1
0 0 1 0
0 0 1 1
0 1 0 0
0 1 0 1
0 1 1 0
0 1 1 1
1 0 0 0
1 0 0 1
1 0 1 0
1 0 1 1
1 1 0 0
1 1 0 1
1 1 1 0
1 1 1 1
```

As shown in the right hand side figure, the sequence of Gray Code is a little bit different from the above. Starting from the second column, 0 appears first and then 1 in the first half but 1 appears first and then 0 in the second half. The bit sequences in the 3rd, 4th column are also 01, 10, 01, 10 in alternating order. Please complete the following recursive program to enumerate the Gray Code:

```
0 0 0 0
0 0 0 1
0 0 1 1
0 0 1 0
0 1 1 0
0 1 1 1
0 1 0 1
0 1 0 0
1 1 0 0
1 1 0 1
1 1 1 1
1 1 1 0
1 0 1 0
1 0 1 1
1 0 0 1
1 0 0 0
```

```
01 #include <stdio.h>
02
03 void gray(int n, int x[], int p) {
04         if (p>=n)
05                 for (int i=0; i<n; i++)
```

```
06                      printf("%d%c", x[i], "\n "[i<n-1]);
07          else {
08
09
10
11          }
12 }
13
14 int main() {
15          int i, n, x[10];
16          while (1==scanf("%d", &n)) {
17                  for (i=0; i<n; i++) x[i]=0;
18                  gray(_______, ________, ________);
19          }
20          return 0;
21 }
```

6. [16] Please complete the following two recursive programs to efficiently calculate $x^n$ using the rule specified below:

(a) $x^n = (x^{n/2})^2$ if $n$ is even; $x^n = x\,(x^{(n-1)/2})^2$ if $n$ is odd

```
01 #include <stdio.h>
02
03 double power1(double x, int n) {
04          double y;
05          if (_____________) // terminating condition
06                  return __________;
07          else if (____________) //    n is odd
08                  return __________ * power1(_________, ___________);
09          else { //    n is even
10                  y = power1(_____________, ___________);
11                  return ___________;
12          }
13 }
14
15 int main() {
16          printf("%f\n", power1(2.5,9));
17          return 0;
18 }
```

(b) $x^n = (x^2)^{n/2}$ if $n$ is even; $x^n = x\,(x^2)^{(n-1)/2}$ if $n$ is odd

```
01 #include <stdio.h>
02
03 double power2(double x, int n) {
04          if (…) // the same as in part (a)
05                  … // the same as in part (a)
06          else if (…) // the same as in part (a)
07                  … // the same as in part (a)
08          else
09                  return power2(____________, _____________);
10 }
11
12 int main() {
13          printf("%f\n", power2(2.5,9));
14          return 0;
15 }
```