1121 NTOUCSE 程式設計 1C Midterm Exam

姓名：________________ 系級：________________ 學號：________________ 112/10/31 (二)

考試時間：**13:20 – 16:00**

Exam rules：1. **Close book**, **close** everything including quizzes, 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. What are the outputs when the following programs are executed on a computer:

(a) [5]
```
int x=3, y=6;
x += y-->5 && x%2==1;
printf("x=%d y=%d", x, y);
```

(b) [5]
```
int x=3, y=5;
x += x/2>1 && (y*=2);
printf("x=%d y=%d", x, y);
```

(c) [5]
```
int i, a[] = {1, 3, 5, 7, 2, 4, 6, 0};
for (i=2; i<5; i++)
    printf("%d ", a[a[a[i]]]);
```

(d) [5]
```
int i, a[] = {1, 3, 5, 7, 2, 4, 6, 0};
int j, t, n = sizeof(a)/sizeof(int);
for (i=0; i<n-1; i++) {
    for (j=n-1; j>i; j--)
        if (a[j-1]<a[j])
            t=a[j], a[j]=a[j-1], a[j-1]=t;
    printf("%d ", a[i]);
}
```

(e) [5]
```
int f(int a[], int i, int j) {
    if (i>j)
        return 0;
    return f(a, i, j-2)*8 + a[j];
}

int main() {
    int a[] = {1, 3, 5, 7, 2, 4, 6, 0};
    printf("%d", f(a, 2, 7));
    return 0;
}
```

2. Please read the following codes and answer the corresponding questions:

```
#include <stdio.h>

int balance = 0;
void save(int amount) {
    balance += amount;
    printf("%d ", balance);
}

int main() {
```

```
10      int i, balance=0;
11      for (i=30; i<100; i+=30)
12          save(i);
13      printf("balance=%d\n", balance);
14      return 0;
15 }
```

(a) Which variables are global variables? [2] What features characterize a global variable?[3]

(b) [5] What is the output on the screen from this program?

(c) [5] Why does the code in line 13 not printing the same value as the code in line 6? How do you modify the variable definition in the function main() such that line 13 does print out the same value as line 6?

(d) [5] Please modify the answer of part (c), use function parameters/arguments and local variables to avoid the usage of global variable in the above codes snippet?

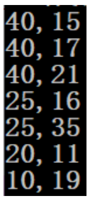
(e) [5] What are the benefits for this program to use global variables?

(f) [5] What are the reasons that nobody in the professional software industry suggest the usage of global variables? or even tolerates the usage of global variables?

3. [15] The two dimensional integer array int points[10][2]; in the following program store the x- and y-coordinates of multiple points on a plane. Please use the qsort() function in the stdlib library to finish the follow program, such that the data in the points array are sorted in descending order according to their x-coordinates, and sorted in ascending order according the their y-coordinates when their x-coordinates coincide (You can write your answers on the answer sheet with the line number of the following program or you can put your answers directly on the problem sheet and make a note on the answer sheet)

```
01 #include <stdio.h>
02 #include _______________
03
04 int compare(________________ a, _________________ b) {
05      int *a1 = (int *)a, *b1=(int *)b;
06      return _____________ ? _______________ : _________________;
07 }
08
09 int main() {
10      int points[][2] = {{40,17}, {10,19}, {25,35}, {40,21}, {40,15}, {20,11}, {25,16}};
11      int i, n=sizeof(points)/sizeof(int[2]);
12      qsort(_____________, ______________, ________________, compare);
13      for (i=0; i<n; i++)
14          printf("%d,%d\n", points[i][0], points[i][1]);
15      return 0;
16 }
```

The outputs of the program are as follows:



```
40, 15
40, 17
40, 21
25, 16
25, 35
20, 11
10, 19
```

4. [20] A d-digit integer $N = (n_1 n_2 n_3 \ldots n_d)_b, n_i \in \{0,1,2,\ldots,b-1\}$ with representation in a number system with base b is a Narcissistic number if the relation $N = {n_1}^d + {n_2}^d + \ldots + {n_d}^d$ is satisfied. The 3-digit decimal integer $(153)_{10} = 1^3 + 5^3 + 3^3$, the 3-digit base-3 integer $(122)_3 = 1^3 + 2^3 + 2^3$, or the 4-digit base-5 integer $(3134)_5 = 3^4 + 1^4 + 3^4 + 4^4 = 419$ are examples of Narcissistic numbers. The 4-digit decimal integer $(1321)_{10}$ is not a Narcissistic number because $1^4 + 3^4 + 2^4 + 1^4 = 99 \neq 1321$. Please complete the following program to determine if a given integer is a Narcissistic number. Each line of the input is a test case containing the base b, b<=10, of the number system and the d-digit number, d<100, the output is "YES" if a number is a Narcissistic number and "NO" otherwise. (You can write your answers on the answer sheet with the line number of the following program or you can put your answers directly on the problem sheet and make a note on the answer sheet)

```
01 #include <stdio.h>
02
03 int main() {
04       int b, i, j, d, z, zd, sum, x;
05       char num[100];
06       while (2==scanf("%d%s", &b, num)) {
07            d=0; while (num[d]_______0) ________; // calculate number of digits
08            for (x=sum=i=0; i<d; i++) {
09                 z = num[i]-________;
10                 x = x*______ + _______; // calculate the value of x in base b
11                 for (zd=z,j=1; ________; j++) zd *= z; // zd holds the d-th power of each digit z
12                 sum += ___________; // calculate the sum of powers
13            }
14            printf(_________ ? "YES\n" : "NO\n");
15       }
16       return 0;
17 }
```

The program in line 11 is a loop to calculate the d-th power of z. It iterates for d-1 times and is considered in efficient. The following is an efficient replacement with a loop iterating only log d times

```
11              for (zd=1, dp=d; dp>0; dp/=2, z*=__________)
12                   if (_______________) zd *= z;
```

In the above code, variable zd is used to hold the d-th power of z, the number of iterations is controlled by the value of dp, initially holding the target power d and divided by 2 after each iteration. Take d=13, z=3 for example, the value to calculate is $z^d = 3^{13} = 3^{2^0 \cdot 1 + 2^1 \cdot 0 + 2^2 \cdot 1 + 2^3 \cdot 1} = (3)^1 \cdot (3^2)^0 \cdot ((3^2)^2)^1 \cdot$

$(((3^2)^2)^2)^1$. This loop works as though it is converting the exponent 13 to its binary representation, bit by bit starting from the lowest one, accompanying by the calculated weights holding in variable z 3, 3^2, $(3^2)^2$, $((3^2)^2)^2 \ldots$ to calculate the target value 3^{13}.

5. (a) [5] Please give an explanation on the actual output of the following program being 17000000.000000 instead of 17000001.000000 ?

```
float x=17000001;
printf("x=%f\n", x);
```

(b) [5] Continuing on problem (a), what is the output of the following program? How many times does the loop actually iterate? Explain your reasoning briefly.

```
float x;
int y;
for (y=x=17000000; x <17000010; x+=1)
    if (!y--) break;
printf("x=%.0f y=%d\n", x, y);
```