計算機程式設計 113 學年度第 2 學期小考 2 試題

系級：_____ 學號：_____ 姓名：_____

All programs have "#include <stdio.h>". Answer **'unknown'** if the value is uncertain.

1. Fill in the space to complete the code.

(1)_____1_____ (2)_____>_____ (3)____a[j]_____

Output	12 34 88 57 9 43 72 12 34 88 57 9 43 72 12 34 57 88 9 43 72 9 12 34 57 88 43 72 9 12 34 43 57 88 72 9 12 34 43 57 72 88

```
01 void print(int a[], int n) {
02    for (int i = 0; i < n; i++)   printf("%d ", a[i]);
03    printf("\n");
04 }
05 void insertionSort(int a[], int n) {
06    int target, i, j;
07    for (i = ___(1)___; i < n; i++) {
08       target = a[i];
09       j = i;
10       while (j > 0 && a[j - 1] ___(2)___ target) {
11          a[j] = a[j - 1];
12          j--;
13       }
14       ___(3)___ = target;
15       print(a, n);
16    }
17 }
18 void main() {
19    int a[] = {34, 12, 88, 57, 9, 43, 72};
20    int n = sizeof(a) / sizeof(a[0]);
21    insertionSort(a, n);
22 }
```

2. Output after executing the following code.

(1)_____6_____ (2)_____8_____ (3)_____12_____

```
01 void func1(int *x, int *y, int *z, int d) {
02    int *temp = y;
03    *z = (*x) + (*temp);
04    *y = *z - (*x);
05    *x = *y + 2;
06    z = temp;
07    x = z;
08    *x = *y * 2;
09    d = *x + *y + *z;
10 }
11 void main() {
12    int a = 3, b = 4, c = -1, d = 5, *m = &a, *n = &b;
13    func1(m, n, &c, d);
14    printf("%d\n", a);       // (1)
15    printf("%d\n", b);       // (2)
16    printf("%d\n", c + d);   // (3)
17 }
```

3. Line 01, 02 are correct. Write True/False and the reason.

(1) Line 03 error (True/False):___ False_____

(2) Line 04 error (True/False):_True. It is not possible to assign a char to a char pointer.(char *)_

(3) Line 05 error (True/False):___ False_____

```
01 void test() {
02    char x[] = "NTUT", y = 'A', *p, *q;
03    p = x;
04    q = y;
05    x[0] = *q;
06 }
```

4. Output after executing the following code.

(1)_____CSIE_____ (2)_____IE_____

```
01 void test() {
02    char s[] = "CSIE", *x = s+1, *y;
03     y = x+1;
04    x[0] = *x; x[1] = *y;
05    printf("%s\n", s);    //(1)
06    printf("%s\n", y);    //(2)
07 }
```

5. Output after executing the following code.

(1)_____1_____ (2)_____4_____ (3)_____5_____

```
01 #include <stdlib.h>
02 void main() {
03    int *p;
04    int *num = (int *)malloc(sizeof(int) * 5);
05    for (int i = 0; i < 5; i++) *(num + i) = i + 1;
06    p = num;
07    printf("%d\n", (*p)++);  //(1)
08    p++;
09    (*p)++;
10    printf("%d\n", ++(*p));  //(2)
11    ++(*p);
12    *(p++);
13    printf("%d\n", num[1]);  //(3)
14 }
```

6. Output after executing the following code.

(1)___unknown____ (2)___ unknown ___ (3)___ unknown ___

```
01 int main() {
02    int a = 10;
03    int *p = &a;
04    int **pp = &p;
05
06    printf("%d\n", pp);    // (1)
07    printf("%d\n", *pp);   // (2)
08    printf("%d\n", p);     // (3)
09    return 0;
10 }
```

7. Output after executing the following code.

(1)_____3_____ (2)_____2_____ (3)_____-2_____

```
01 #include <stdlib.h>
02 int main() {
03    int i = 1, j, number = 2, *p = &number;
04    for (j = 0; j < number; j++) (*p) += i--;
05    printf("%d\n", j);          //  (1)
06    printf("%d\n", number); //  (2)
07    printf("%d\n", i);          //  (3)
08 }
```

8. Output after executing the following code.

	5	6	1	3	8	10	9
(1)	1	3	5	6	8	10	9
(2)	1	3	5	6	8	10	9
	1	3	5	6	8	9	10

```
01 #define SWAP(x, y) {int t; t = x; x = y; y = t;}
02 void printData(int data[]) {
03    for (int i=0; i<7; i++) printf("%d, ", data[i]);
04    printf("\n");
05 }
06 void QuickSort(int data[], int left, int right) {
07    int i, j, target;
08    if (left >= right) return;
09    i = left;
10    j = right;
11    target = data[left];
12    while (i != j) {
13       while ((data[j] > target) && (i < j)) j--;
14       while ((data[i] <= target) && (i < j)) i++;
15       if (i < j) SWAP(data[i], data[j]);
16    }
17    SWAP(data[left], data[i])
18    printData(data);
19    QuickSort(data, left, i - 1);
20    QuickSort(data, i + 1, right);
21 }
22 void main() {
23    int arr[] = {8, 6, 1, 10, 5, 3, 9};
24    QuickSort(arr, 0, 6);
25 }
```

9. score[1] = ___undefined____

```
01 int score[10] = {1};
```

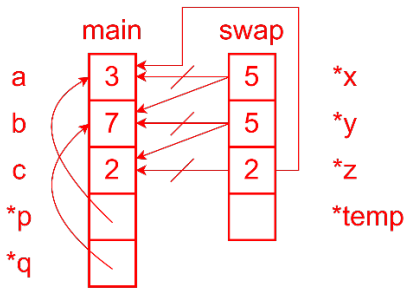
10. Output after executing the following code.

(1)_ $$$$$ation _ (2)_#########l _ (3)_ ################ed_

```
01  #include <string.h>
02  void main() {
03      char s1[] = "graduation";
04      char s2[][7] = {"happy", "Hel\0lo", "red"};
05      printf("%s\n", memset(s1, '$', 5));
06      printf("%s\n", memset(s2, '#', 9));
07      printf("%s\n", memset(s2, '#', 15));
08  }
```

11. Please trace the program and draw the diagram of the relationship between main and swap function. (Including variables, parameters, pointers and their values)

```
01  void swap(int *x, int *y, int *z) {
02      int *temp = x;
03      *y = (*z) + (*temp);
04      *z = (*y) - (*temp);
05      *x = 5;
06      x = y;
07      y = z;
08      z = temp;
09  }
10
11  int main(void) {
12      int a = 3, b = 7, c = 2, *p = &a, *q = &b;
13      swap(p, q, &c);
14      printf("%d %d %d\n", a, b, c);
15      return 0;
16  }
```



2. Output after executing the following code.

(1)_____14_____ (2)_____98_____ (3)____8_____

```
01  void f1(int **x, int **y, int **z, int *w) {
02      *x = w;
03      **y = (**x) + (**z);
04      *w = (**y) * (**x);
05  }
06
07  void g1(int **x, int **y, int **z, int *w) {
08      int **temp = x;
09      x = z;
10      z = y;
11      y = temp;
12      *x = w;
13      (**x) = (**y) - (**z);
14  }
15  void main() {
16      int a = 4, b = -3, c = 7, *p = &a, *q = &b, *r = &c;
17      f1(&p, &q, &r, &c);
18      printf("%d\n", b); // (1)
19      printf("%d\n", c); // (2)
20      p = &a, q = &b, r = &c;
21      g1(&p, &q, &r, &c);
22      printf("%d\n", a + b + c); // (3)
23  }
```

13. Please describe your learning problem and how to improve it. (30 or more word will be scored)

14. After executing bubbleSort({64,34,25,12,22,11,90}, 7):

The output is:

	25	12	22	11	34	64	90
(1)	12	22	11	25	34	64	90
(2)	12	11	22	25	34	64	90
	11	12	22	25	34	64	90
	11	12	22	25	34	64	90
(3)	swap_count =				14		

```
01  void bubbleSort(int arr[], int n) {
02      int swap_count = 0, temp;
03      for (int i = 0; i < n - 1; i++) {
04          for (int j = 0; j < n - i - 1; j++) {
05              if (arr[j] > arr[j + 1]) {
06                  temp = arr[j];
07                  arr[j] = arr[j + 1];
08                  arr[j + 1] = temp;
09                  swap_count = swap_count + 1;
10              }
11          }
12      }
13      printf("swap_count = %d\n", swap_count);
14  }
```

15. Complete the diagram based on the following code.

```
01  void copy(int C[], int a[], int m, int n) {
02      for (int i = m; i <= n; i++) a[i] = C[i];
03  }
04  void merge(int C[], int A[], int am, int an, int B[], int bm, int bn)
05      int k = am;
06      while ((am <= an) && (bm <= bn)) {
07          if (A[am] <= B[bm])
08              C[k++] = A[am++];
09          else
10              C[k++] = B[bm++];
11      }
12      while (am <= an) C[k++] = A[am++];
13      while (bm <= bn) C[k++] = B[bm++];
14  }
15  void mergeSort(int a[], int m, int n) {
16      int mid = 0, C[20];
17      if (n > m) {
18          mid = (m + n) / 2;
19          mergeSort(a, m, mid);
20          mergeSort(a, mid + 1, n);
21          merge(C, a, m, mid, a, mid + 1, n);
22          copy(C, a, m, n);
23      }
24  }
```