

Discussion time: 5th November 2022, 22:57-23:04

The screenshot shows a Zoom meeting interface. On the left, a C++ code editor displays a bubble sort implementation. The code is as follows:

```
10 {
11     for (j = 0; k = 0; j < SIZE-1-i; j++)//a要求是加入-i
12     {
13         if (a[j] > a[j + 1])//向上浮一個
14         {
15             tmp = a[j];
16             a[j] = a[j + 1];
17             a[j + 1] = tmp;
18         }
19         else//b要求，都沒有改變則跳出
20         {
21             k++;
22             if (k == SIZE-1-i) goto end;
23         }
24     }
25     printf("Loop %d :", i+1);
26     for (j = 0; j < SIZE; j++)
27         printf("%5d", a[j]);
28     printf("\n");
29 }
```

On the right, three participants are visible in a vertical stack. The top participant is a man with glasses, the middle is a man with a white shirt, and the bottom is a man with glasses. The Zoom status bar at the bottom indicates the time is 10:59 and the user is beebnq-ew.

The screenshot shows a Zoom meeting interface. On the left, a PDF document titled "NTUT-CL-CH4.pdf" is displayed. It contains a C++ code snippet for bubble sort and a table illustrating the array state at each iteration.

氣泡排序 (bubble sort)

```
1 #include <stdio.h>
2 #include <stdlib.h>
3
4 void main()
5 {
6     int i, j, tmp;
7     int a[5] = {26, 5, 81, 7, 63};
8     for (i = 0; i < 4; i++)
9     {
10         for (j = 0; j < 4-i; j++)
11         {
12             if (a[j] > a[j+1])
13             {
14                 tmp = a[j];
15                 a[j] = a[j+1];
16                 a[j+1] = tmp;
17             }
18         }
19         printf("Loop %d:", i);
20         for (j = 0; j < 5; j++)
21             printf("%5d", a[j]);
22         printf("\n");
23     }
24     printf("pass\n");
25 }
```

Table illustrating array state:

	a[0]	a[1]	a[2]	a[3]	a[4]
Initial	26	5	81	7	63
j=0	5	26	81	7	63
j=1	5	26	81	7	63
j=2	5	26	7	81	63
j=3	5	26	7	63	81
j=0	5	26	7	63	81
j=1	5	7	26	63	81
j=2	5	7	26	63	81
j=3	5	7	26	63	81

On the right, three participants are visible in a vertical stack. The Zoom status bar at the bottom indicates the time is 11:02 and the user is beebnq-ew.

The screenshot shows a Zoom meeting interface. On the left, a C++ code editor displays a bubble sort implementation. The code is as follows:

```
7 int i, j, k, tmp;
8 int a[SIZE] = { 1254, 77, 8, 774, 54 };
9 for (i = 0; i < SIZE-1; i++)
10 {
11     for (j = 0; k = 0; j < SIZE-1-i; j++)//a要求是加入-i
12     {
13         if (a[j] > a[j + 1])//向上浮一個
14         {
15             tmp = a[j];
16             a[j] = a[j + 1];
17             a[j + 1] = tmp;
18         }
19         else//b要求，都沒有改變則跳出
20         {
21             k++;
22             if (k == SIZE-1-i) goto end;
23         }
24     }
25 }
```

On the right, three participants are visible in a vertical stack. The Zoom status bar at the bottom indicates the time is 11:01 and the user is beebnq-ew.

P04

```
main.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      int n[10], i;
7
8      for (i = 0; i < 10; i++)
9      {
10         n[i] = 0;
11     }
12
13     printf("%s%10s\n", "Element", "Value");
14
15     for (i = 0; i < 10; i++)
16     {
17         printf("%7d%10d\n", i, n[i]);
18     }
19
20     system ("pause");
21     return 0;
22 }
23
```

P05

```
main.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      int n[10] = {15,14,25,74,26,82,58,24,85,24};
7      int i;
8
9      printf("%s%10s\n", "Element", "Value");
10
11     for (i = 0; i < 10; i++)
12     {
13         printf("%7d%10d\n", i, n[i]);
14     }
15
16     system ("pause");
17     return 0;
18 }
19
```

P06

```
main.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define SIZE 10
4
5  int main()
6  {
7      int n[SIZE];
8      int i;
9
10     for (i = 0; i < SIZE; i++)
11     {
12         n[i] = 2+2*i;
13     }
14
15     printf("%s%10s\n", "Element", "Value");
16
17     for (i = 0; i < SIZE; i++)
18     {
19         printf("%7d%10d\n", i, n[i]);
20     }
21
22     system ("pause");
23     return 0;
24 }
25
```

P07

```
main.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define SIZE 10
4
5  int main()
6  {
7      int n[SIZE] = {15,14,25,74,26,82,58,24,85,24};
8      int i;
9      int total = 0;
10
11     for (i = 0; i < SIZE; i++)
12     {
13         total += i[n];
14     }
15
16     printf("Total of array element value is: %d\n", total);
17
18     system ("pause");
19     return 0;
20 }
21
```

P08

```
main.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define SIZE 10
4
5  int main()
6  {
7      int n[SIZE] = {10,1,9,2,8,3,7,4,6,5};
8      int i, j;
9
10     printf("%s%10s%15s\n", "Element", "Value", "Histogram");
11
12     for (i = 0; i < SIZE; i++)
13     {
14         printf("%7d%10d", i, n[i]);
15         for (j = 1; j <= n[i]; j++)
16         {
17             printf("**");
18         }
19         printf("\n");
20     }
21
22     system ("pause");
23     return 0;
24 }
25
```

P13

```
main.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define SIZE 10
4
5  void printer(const int a[][3]);
6
7  int main()
8  {
9      int A[2][3] = {{1,2,3},{4,5,6}};
10     int B[2][3] = {1,2,3,4,5};
11     int C[2][3] = {{1,2},{3}};
12
13     printf("Values is A by row is:\n");
14     printer(A);
15
16     printf("Values is B by row is:\n");
17     printer(B);
18
19     printf("Values is C by row is:\n");
20     printer(C);
21
22     system ("pause");
23     return 0;
24 }
25
26 void printer(const int a[][3])
27 {
28     int i, j;
29
30     for (i = 0; i <= 1; i++)
31     {
32         for (j = 0; j <= 2; j++)
33         {
34             printf("%d ", a[i][j]);
35         }
36         printf("\n");
37     }
38 }
39
```

```

main.c x
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define STUDENT 3
4  #define EXAM 4
5
6  void printer(const int grade[][EXAM], int pupil, int test);
7  int min(const int grade[][EXAM], int pupil, int test);
8  int max(const int grade[][EXAM], int pupil, int test);
9  double avg(const int setgrade[], int test);
10
11 int main()
12 {
13     int student;
14     const int sGrade[STUDENT][EXAM] =
15     {{67,89,90,79},{80,90,94,85},{78,87,98,70}};
16
17     printf("Array:\n");
18     printer(sGrade, STUDENT, EXAM);
19
20     printf("\n\nLowest grade: %d\nHighest grade: %d\n", min(sGrade, STUDENT, EXAM), max(sGrade, STUDENT, EXAM));
21
22     for (student = 0; student < STUDENT; student++)
23     {
24         printf("The average score of student %d is: %.2f\n", student, avg(sGrade[student], EXAM));
25     }
26
27     system ("pause");
28     return 0;
29 }
30
31 void printer(const int grade[][EXAM], int pupil, int test)
32 {
33     int i, j;
34
35     printf("          [0]   [1]   [2]   [3]");
36
37     for (i = 0; i < pupil; i++)
38     {
39         printf("\nsGrade[%d] ", i);
40         for (j = 0; j < test; j++)
41         {
42             printf("%-5d ", grade[i][j]);
43         }
44     }
45 }
46
47 int min(const int grade[][EXAM], int pupil, int test)
48 {
49     int i, j;
50     int low = 100;
51
52     for (i = 0; i < pupil; i++)
53     {
54         for (j = 0; j < test; j++)
55         {
56             if (grade[i][j]<low)
57                 low = grade[i][j];
58         }
59     }
60     return low;
61 }
62
63 int max(const int grade[][EXAM], int pupil, int test)
64 {
65     int i, j;
66     int high = 0;
67
68     for (i = 0; i < pupil; i++)
69     {
70         for (j = 0; j < test; j++)
71         {
72             if (grade[i][j]>high)
73                 high = grade[i][j];
74         }
75     }
76     return high;
77 }
78
79 double avg(const int setgrade[], int test)
80 {
81     int i;
82     int total = 0;
83
84     for (i = 0; i < test; i++)
85     {
86         total += setgrade[i];
87     }
88
89     return (double)total/test;
90 }
91

```

P19

```
main.c x
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      char string1[20];
7      char string2[] = "Not Bruh Moment";
8      int i;
9
10     printf("Enter a string: ");
11     scanf("%s", string1);
12
13     printf("string1 is: %s\nstring2 is: %s\n", string1, string2);
14     printf("string 1 with spaces in between characters is: \n", string1, string2);
15
16     for (i = 0; string1[i] != '\0'; i++)
17     {
18         printf("%c ", string1[i]);
19     }
20     printf("\n");
21     system ("pause");
22     return 0;
23 }
24
```

P21

```
main.c x
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  void Static(void);
5  void Auto(void);
6
7  int main()
8  {
9      printf("First call to each function:\n");
10     Static();
11     Auto();
12
13     printf("\n\nSecond call to each function:\n");
14     Static();
15     Auto();
16
17     printf("\n");
18     system ("pause");
19     return 0;
20 }
21
22 void Static(void)
23 {
24     static int array1[3];
25     int i;
26
27     printf("\nStatic former value:\n");
28
29     for (i = 0; i <= 2; i++)
30         printf("array1[%d] = %d ", i, array1[i]);
31
32     printf("\nStatic latter value:\n");
33
34     for (i = 0; i <= 2; i++)
35         printf("array1[%d] = %d ", i, array1[i]+=5);
36 }
37
38 void Auto(void)
39 {
40     int array2[3];
41     int i;
42
43     printf("\nAuto former value:\n");
44
45     for (i = 0; i <= 2; i++)
46         printf("array2[%d] = %d ", i, array2[i]);
47
48     printf("\nAuto latter value:\n");
49
50     for (i = 0; i <= 2; i++)
51         printf("array2[%d] = %d ", i, array2[i]+=5);
52 }
53
```

P25

```
main.c x
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int add1(int);
5
6  int main()
7  {
8      int x = 100;
9      int y = add1(x);
10     printf("x=%d\n", x);
11     system ("pause");
12 }
13
14 int add1(int xref)
15 {
16     xref++;
17     printf("xref=%d\n", xref);
18     return xref;
19 }
20
```

P26

```
main.c x
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  int main()
5  {
6      char array[5];
7      printf("    array = %p\n&array[0] = %p\n    &array = %d", array, &array[0], &array);
8
9      system ("pause");
10     return 0;
11 }
12
```

```
main.c x
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define SIZE 5
4
5  void modArray (int b[], int size);
6  void modElement(int e);
7
8  int main()
9  {
10     int a[SIZE] = {0,1,2,3,4};
11     int i;
12
13     printf("Effects of passing entire array by reference:\n\n"
14           "The values of the original array are:\n");
15
16     for (i = 0; i < SIZE; i++)
17     {
18         printf("%3d", a[i]);
19     }
20     printf("\n");
21
22     modArray(a, SIZE);
23     printf("The values of the modified array are:\n");
24     for (i = 0; i < SIZE; i++)
25     {
26         printf("%3d", a[i]);
27     }
28     printf("\n");
29
30     printf("Effects of passing array element by value:\n\n"
31           "The values a[3] is: %d\n", a[3]);
32
33     modElement(a[3]);
34     printf("The value of a[3] is: %d\n", a[3]);
35
36     system ("pause");
37     return 0;
38 }
39
40 void modArray(int b[], int size)
41 {
42     int j;
43
44     for (j = 0; j < size; j++)
45         b[j] *= 2;
46 }
47
48 void modElement(int e)
49 {
50     printf("Value in modElement is: %d\n", e *= 2);
51 }
52
```

P30

```
main.c x
1  #include <stdio.h>
2  #include <stdlib.h>
3
4  void inv (int*);
5
6  int main()
7  {
8      int a[3] = {1,2,3};
9      int i;
10
11     for (i = 0; i < 3; i++)
12     {
13         printf("%d ", a[i]);
14     }
15     printf("\n");
16
17     inv(a);
18
19     for (i = 0; i < 3; i++)
20     {
21         printf("%d ", a[i]);
22     }
23     printf("\n");
24
25     system ("pause");
26     return 0;
27 }
28
29 void inv(int *b)
30 {
31     int temp[3], i;
32     for (i = 0; i < 3; i++)
33         temp[2-i] = b[i];
34
35     for (i = 0; i < 3; i++)
36         b[i] = temp[i];
37 }
38
```

P37

```
main.c x
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define SIZE 100
4
5  int linear(const int array[], int key, int size);
6
7  int main()
8  {
9      int a[SIZE];
10     int x;
11     int searchKey;
12     int element;
13
14     for (x = 0; x < SIZE; x++)
15         a[x] = 2*x;
16
17     printf("Enter integer search key:\n");
18     scanf("%d", &searchKey);
19
20     element = linear(a, searchKey, SIZE);
21
22     if (element != -1)
23     {
24         printf("Found value in element %d\n", element);
25     }
26
27     else
28     {
29         printf("Value not found!\n");
30     }
31
32     system ("pause");
33     return 0;
34 }
35
36 int linear(const int array[], int key, int size)
37 {
38     int n;
39     for (n = 0; n < size; ++n)
40     {
41         if (array[n] == key)
42         {
43             return n;
44         }
45     }
46     return -1;
47 }
48
```


6.11

```
main.c X
1  #include <stdio.h>
2  #include <stdlib.h>
3  #define SIZE 5
4
5  int main()
6  {
7      int i, j, k, temp;
8      int a[SIZE] = {26, 63, 7, 81, 5};
9
10     for (i = 0; i < SIZE - 1; i++)
11     {
12         for (j = 0, k = 0; j < SIZE - 1 - i; j++)
13         {
14             if (a[j] > a[j+1])
15             {
16                 temp = a[j];
17                 a[j] = a[j+1];
18                 a[j+1] = temp;
19             }
20             else
21             {
22                 k++;
23                 if (k == SIZE - 1 - i) goto Bruh;
24             }
25         }
26         printf("Loop %d: ", i);
27         for (j = 0; j < 5; j++)
28             printf("%4d", a[j]);
29         printf("\n");
30     }
31     Bruh:
32     system ("pause");
33     return 0;
34 }
35
36
```

Conclusion:

The usage of arrays is very useful if you are using a lot of values. The usage of call by value, call by address, or even call by reference can simplify a program by a lot, especially when you no longer need to assign a function to a data just because you need to. Static arrays are also useful if you need to store values.

Code: <https://github.com/AldrichWijaya/Homework.git>