SouthWest University

Lab report

Couse name C programming

Semester 2019 - 2020 - 2

Grade 2019 class Software Engineering 3

student Reg. No 222019321062074

Tutor James Chen

School of Computer and Information Science

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Lab 2 | | | **Practicing on arrays** | | | | | |
| Issue Date | | 2020年3月25日 | | | experimental types | | □validation experiment,  □comprehensive experiment  ☑design experiment | |
| **Goal**  • You will practice using arrays  **Assignment**  1.A 1-D Array has ten elements. Please use bubble sorting method to sort the array in descending order.  2**.** Find if there’s a saddle point in a matrix. If there is one, output its position in the matrix. If not, output corresponding information. (In a matrix, if there’s one element which is the largest in its row and smallest in its column, then it’s called a saddle point.  3. Write a program to output a Yanghui Triangle.  4. Write a program that can count the number of occurrences of uppercase letters, lowercase letters, digits, blanks and other characters in an article which has three lines and at most 80 characters at each line. | | | | | | | | |
| * experimental contents and process  1. Analyze what the problems say, and understand what it means. 2. Clear object procedure. 3. Compilation object process. 4. Make the program functional. | | | | | | | | |
| * screen shots of the VC++ or other IDE for C(Dev c++，VS2010. etc)showing the output results of running your Lab code.   Ex1：    Ex2：    Ex3：    Ex4： | | | | | | | | |
| * Lab Code   Ex1：  #include <stdio.h>  int a[11], i, j, tmp;  void bubble\_sort(void)  {  for (i = 0; i < 10; i++)  for (j = 0; j < 9 - i; j++)  if (a[j+1] < a[j])  {  tmp = a[j+1];  a[j+1] = a[j];  a[j] = tmp;  }  }  int main (void)  {  for (i = 0; i < 10; i++)  scanf("%d", &a[i]);  bubble\_sort();  for (i = 0; i < 10; i++)  printf("%d ", a[i]);  return 0;  }  Ex2：  #include <stdio.h>  int main (void)  {  int n, a[6][6], i, j, k, flagrow, flagcol, flag = 0;  scanf("%d", &n);  for (i = 0; i <= n - 1; i++)  for (j = 0; j <= n - 1; j++)  scanf("%d", &a[i][j]);  for (i = 0; i <= n - 1; i++)  for (j = 0; j <= n - 1; j++)  {  flagrow = 0;  flagcol = 0;  for (k = 0; k <= n - 1; k++)  if (a[i][j] < a[i][k])  flagrow = 1;  for (k = 0; k <= n - 1; k++)  if (a[i][j] > a[k][j])  flagcol = 1;  if (!flagrow && !flagcol)  {  printf("%d %d", i, j);  return 0;  }  }  printf("NONE");  return 0;  }  Ex3：  #include <stdio.h>  int main (void)  {  int a[11][11] = {0}, i, j, n;  a[1][1] = 1;  a[2][1] = 1;  a[2][2] = 1;  for (i = 3; i <= 10; i++)  for (j = 1; j <= i; j++)  a[i][j] = a[i-1][j-1] + a[i-1][j];  scanf("%d", &n);  for (i = 1; i <= n; i++)  {  for (j = 1; j <= n - i; j++)  printf(" ");  for (j = 1; j <= n; j++)  if (a[i][j])  printf("%4d", a[i][j]);  printf("\n");  }  return 0;  }  Ex4：  #include <stdio.h>  #include <string.h>  int main (void)  {  char c[4][81];  int upper = 0, lower = 0, num = 0, space = 0, others = 0, i, j;  for (i = 1; i <= 3; i++)  gets(c[i]);  for (i = 0; i <= 3; i++)  for (j = 0; j <= strlen(c[i]); j++)  if (c[i][j] >= 48 && c[i][j] <= 57)  num++;  else if (c[i][j] >= 65 && c[i][j] <= 90)  upper++;  else if (c[i][j] >= 97 && c[i][j] <= 122)  lower++;  else if (c[i][j] == 32)  space++;  else  others++;  printf("Uppercase letters:%d\nLowercase letters:%d\nDigits:%d\n", upper, lower, num);  printf("Blanks:%d\nother characters:%d", space, others);  return 0;  } | | | | | | | | |
| * experimental summary/Analysis   Through this experiment, I got a better understanding of array and its usage, knew arrays are stored continuously in memory, mastered to use loop-structure to help input or output an array.  In ex1, sorting is an important algorithm, this experiment I knew two kinds of fundamental sorting algorithm: bubble sort and selection sort, both of them are easy to handle.  In ex2,3 and 4 I used 2 dimensions array to store a matrix or sentences, which helps a lot during calling it.  In ex4, I try to combine characters and ascii code to execute operations, and use string function ‘strlen’ in <string.h> used as loop condition. | | | | | | | | |
|  | Criteria | | | | | | | scale |
| Goal | | | | | | | A B C D E |
| Process | | | | | | |
| Design | | | | | | |
| Algorithm | | | | | | |
| Code | | | | | | |
| Data/Results | | | | | | |
| summary | | | | | | |
| written | | | | | | |
| Score | | |  | | tutor Signature：  Date: : | | |
| * Lab Evaluation Criteria   A: This lab is exceptional, working and meeting all of the specifications.The code is exceptionally well organized and very easy to follow.The code could be reused as a whole or each routine could be reused.The documentation is well written and clearly explains what the code is accomplishing and how.The program was delivered on time.The code is extremely efficient without sacrificing readability and understanding.  B: This lab is very good--works and produces the correct results and displays them correctly. It also meets most of the other specifications. The code is fairly easy to read. Most of the code could be reused in other programs. The documentation consists of embedded comment and some simple header documentation that is somewhat useful in understanding the code. The program was delivered within a week of the due date. The code is fairly efficient without sacrificing readability and understanding.  C: This lab is adequate, with only minor deficiencies. The program produces correct results but does not display them correctly. The code is readable only by someone who knows what it is supposed to be doing. Some parts of the code could be reused in other programs. The documentation is simply comments embedded in the code with some simple header comments separating routines. The code was within 2 weeks of the due date. The code is brute force and unnecessarily long..  D: This lab shows some effort but has at least one major deficiency.The program is producing incorrect results. The code is poorly organized and very difficult to read. The code is not organized for reusability. The documentation is simply comments embedded in the code and does not help the reader understand the code. The code was more than 2 weeks overdue. The code is huge and appears to be patched together.  E: This lab is poorly written and shows very little effort or understanding. | | | | | | | | |