复旦大学计算机科学技术学院

《程序设计》期中考试试卷

共 5页

课程代码：COMP120006 考试形式：□开卷 □√闭卷 2023 年 11月10日

（本试卷答卷时间为120分钟，答案必须写在答题纸上，做在试卷上无效）

专业 学号 姓名

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| 题号 | 一 | 二 | 三 | 四 | 五 | 六 | 总分 |
| 分值 |  |  |  |  |  |  | 100 |

一、单选题（每题2.5分，共20分）

1、请指出以下可以作为C语言用户标识符（ ）

A．while B．for

C．\_x1y2z3 D．2a

2、有数组定义：int *a* [][3] = {1, 2, 3, 4, 5}; 则第一维的元素个数为( )

A．2 　　　B．3 　　C．4 　　　　D．无确定值

3、有如下程序段，

int a, b =3, c=4, d = 5;

a = ++b, c--, d+3;

运行之后，a的值是( )

A、4 B、 3 C、 8 D、 5

b的值是( )

A、4 B、 3 C、 8 D、 5

整个逗号表达式的值是( )

A、4 B、 3 C、 8 D、 5

4、有如下程序

#include <stdio.h>

void main()

{

int n;

for(n=100; n<=200; n++){

if(n%3==0) {

printf("%d ",n); break;

}

}

printf("\n");

}

该程序的输出结果是( )

A、100 B、101 C、102 D、103

5、以下程序执行后b的值是：( )

#include <stdio.h>

int main( )

{

short int a, b;

a=32767;

b=a+1;

return 0;

}

1. 32768 B、-32768 C)、0 D、不确定

二、程序阅读题（每题5分，共20分）

1、以下程序运行所产生的输出是\_\_ The upper case is x\_\_\_\_\_\_\_\_。

#include <stdio.h>

int main()

{

char c1, c2;

c1='X';

c2=c1+32;

printf("The upper case is %c\n",c2);

return 0;

}

2、以下程序运行所产生的输出是\_ a+u=22, b+u=-14\_\_\_\_\_。

#include <stdio.h>

int main( )

{

int a, b, c, d;

unsigned u;

a=12; b=-24; u=10;

c=a+u; d=b+u;

printf("a+u=%d, b+u=%d",c, d);

return 0;

}

1. 请阅读下面程序段，阐述程序段的功能。

int a[100]={0};

int n;

string s;

cin>>s; //输入数字字符串

n=s.length(); //计算数字字符串长度

for (i=0; i<n; i++)

a[i]=s[n-i-1]-'0';

用数字字符串来接收数据，然后将字符串中的各位字符转换为对应的十进制数，并按十进制位的顺序存储到一个数组中。

1. 在大于1的自然数中，只能被1和这个数本身整除的数被称为素数，又叫质数。定义整数数组*u*[]和*prime*[]，请阅读下面的程序段；阐述程序段的功能；并说明整数数组*u*[]和*prime*[]的作用。

int *i*, *j*, *n, num*=0;

scanf("%d", &*n*);

for (*i*=2; *i*<=*n*; *i*++) *u*[*i*]=1;

for (*i*=2; *i*<=*n*; *i*++)

if (*u*[*i*])

for (*j*=2; *j\*i*<=*n*; *j*++)

*u*[*j\*i*]=0;

for (*i*=2; *i*<=*n*; *i*++) if (*u*[*i*])

*prime*[++*num*]=*i*;

计算整数区间[2，*n*]中的所有素数，对于数组*u*[]，*u*[*i*]=1，*i*为素数，*u*[*i*]=0，*i*为合数。[2，*n*]中的所有素数，从小到大，存储在数组*prime*[ ]中。

三、完全程序题（每空3分，共30分）

1、直接插入排序，就是将一个记录插入到已排好序的有序表中，从而得到一个新的、记录数量增1的有序表。通过构建有序序列，对于未排序数据，在已排序序列中从后向前扫描，找到相应位置并插入。

例如，对于原始序列：2， 4 ，1， 3， 5；插入排序的过程如下：

第一轮，得序列：2；

第二轮，插入4，得序列：2， 4；

第三轮，插入1，得序列：1， 2 ，4；

第四轮，插入3，得序列：1， 2 ，3， 4；

第五轮，插入5，得序列：1， 2 ，3， 4， 5；

插入排序的C语言程序段如下：

int arr[], len; // len为数组arr中要排序的元素个数

for(int i=1; i<len; i++) { // 从下标为1的元素开始选择合适的位置插入

int key=arr[i]; // 记录要插入的数据

int j=i-1;

while((j>=0) && (key<arr[j])){ // 从已经排序的序列最右边的开始比较，找到比其小的数

arr[j+1]=arr[j];

j--;

}

arr[j+1]=key; //存在比其小的数，插入

}

2、选择排序是最简单的排序算法之一：第一次从待排序的数据元素中选出最小（或最大）元素，存放在序列的起始位置；然后，再从剩余的未排序元素中寻找到最小（大）元素，放到已排序的序列的末尾。以此类推，直到全部待排序的数据元素的排列好。

例如，对于原始序列：2， 4 ，1， 3， 5；选择排序的过程如下：

第一轮，找到最小值，和第1个元素交换，得序列：1， 4 ，2， 3， 5；

第二轮，找到次小值，和第2个元素交换，得序列：1， 2 ，4， 3， 5；

第三轮，找到第三小的值，和第3个元素交换，得序列：1， 2 ，3， 4， 5；

以此类推，最后得到序列：1， 2 ，3， 4， 5。

选择排序C语言程序段如下。

int arr[]; int len; // len为数组arr中要排序的元素个数

int i, j, temp;

for (i = 0 ; i < len - 1 ; i++)

{

int min = i;

for (j = i + 1; j < len; j++) //遍历未排序的元素

if (arr[j] < arr[min]) //找到目前最小值

min = j; //记录最小值的位置

temp = arr[min]; arr[min] = arr[i]; arr[i]= temp; //做互换

}

}

3、设T和P是两个字符串(T的长度为n，P的长度为m，1≤m≤n)，要求在T中查找是否有与P相等的子串。如果有，则给出P在T中的匹配位置，这个运算被称为模式匹配。

实现模式匹配的Brute Force算法如下：

①按顺序遍历T的字符串，将T的每个字符作为匹配的起始字符，用P的字符依次与T中的字符做比较，判断是否匹配。

②如果T和P的当前字符匹配，则比较下一个字符；否则，以T的当前起始字符的下个字符为起始字符，重复上述步骤。

③如果T的一个子字符串与P匹配，则返回T中的匹配子串的起始字符的序号，匹配成功；如果在T中不存在与P匹配的子字符串，则匹配失败。

实现Brute Force算法的程序段如下：

m = strlen(P);

n = strlen(T);

for (i = 0; i <= n - m; ++i)

{

j = 0;

while (j < m && T[i + j] == P[j])

++j;

if (j == m)

return i;

}

四、编程题（每题15分，共30分）

其中90％为程序正确性评分，10％为程序其他质量指标评分（可读性、书写风格等）。所有计算过程中的数据视为始终在int范围内。

1、矩阵*A*和矩阵*B*相乘的前提条件是矩阵*A*的列数和矩阵*B*的行数相等，相乘的结果矩阵的行数等于矩阵*A*的行数，列数等于矩阵*B*的列数；即，设*A*为*m*×*p*的矩阵，*B*为*p*×*n*的矩阵，则*m×n* 的矩阵*C*为矩阵*A*与*B*的乘积，记作*C=A*×*B*，其中矩阵*C*中的第*i* 行第*j*列元素*cij*==*ai*1*b*1*j*+*ai*2*b*2*j*+…+*aipbpj*。

请您写出矩阵*A*和矩阵*B*相乘的程序，要求，输入矩阵*A*和矩阵*B*的行数和列数，如果可以相乘，则输入矩阵*A*和矩阵*B*，实现*A*和矩阵*B*相乘，输出结果。

2、编写程序，输出1、2、3 、4个数字组成的所有互不相同且无重复数字的三位数。

#include "stdio.h"

main()

{

int i,j,k;

printf("\n");

for(i=1;i<5;i++)

for(j=1;j<5;j++)

for (k=1;k<5;k++)

if (i!=k&&i!=j&&j!=k)

printf("%d,%d,%d\n",i,j,k);

}