2019 级电信学院《计算机与程序设计基础(C)》考试试题（A）

（考试时间：2020.01.06 考试时长：3 小时）

题目编号	1(10')	2(5')	3(10')	4(15')	5(20')	6(15')	7(25')	总分(100')
得分								

一、判断下列语句或程序段的对错。（“×”表示错，“√”表示对）（10 分）

(1) int x = y = z = 100; （×）

(2) int main = (0x18 == 'A'); （√）

(3) float number = 3.92E-0.9; （×）

(4) char array[] = {1, 2, 3, 4, 5, 6}; （√）

(5) int iArray[2][] = {10, 20, 30, 40, 50}; （×）

(6) char str[20];
str = "string"; （×）

(7) int a = 5, b = 7, c;
执行 c = a+++(++b) 语句后，a、b、c 的值分别为 6, 8, 12。 （×）

(8) char *chPtr = NULL;
scanf("%s", chPtr); （×）

(9) int (*aPtr)[8], array[6][8];
aPtr=array+1; （√）

(10) register int n;
scanf("%d", &n); （×）

二、计算表达式的值（假定各题计算彼此独立）（5 分）

设 **unsigned char** x=3, y=5, a=7, b=14, c=6, d=8; **float** f=1.0; **char result**;

(1) **f = b/x + d%y;** //求 f 的值 (7.0 或者 7)

(2) **!(x > y) && (++a - b)** //求表达式值 (1)

(3) **result = (a & b) ^ (~c | d);** //求 result 的值 (-1)

(4) **a += b %= a+b;** //求 a 的值 (21)

(5) **a++, a - d;** //求运算结果 (0)

密 封 线

成绩_____ 学号_____ 姓名_____ 班级_____ 学院_____

三、单选题（10 分）

(1) 设整型变量 a、b、c 的值均为 2，运行表达式语句“++a && (++ b || ++c)”后 c 的值是(B)

A) 3　　B) 2　　C) 1　　D) 0

(2) 设 char strArray[15] = {"Aa\t12\n**0**is15\n"};则 sizeof(strArray)的值是（ B ）

A) 14　　B) 15　　C)7　　D) 8

(3) char 型常量在内存中存入的是(A)

A) ASCII 代码值　B) BCD 代码值　C) 内码值　D) 十进制代码值

(4) 设有 float f=13.8; int num; 执行语句 **num=((int)f)/3**; 后，变量 num 的值是(B)

A) 1　　B) 4　　C) 4.333333　　D) 4.6

(5) C 语言规定，函数返回值的数据类型是由(D)

A) return 语句中的表达式类型所决定

B) 调用该函数时的主调函数类型所决定

C) 调用该函数时系统临时决定

D) 在定义该函数时所指定的函数类型所决定

(6) 在 C 语言中，数据变量的**缺省存储类型**是(A)

A) auto　　B) static　　C) extern　　D) register

(7) 设 str 是一个字符串数组(**char str[80];**)，下面哪一条语句的执行结果和其他语句的执行结果是**不同的**(D)

A) *str=0;　　B) str[0]= '\0';　　C) strcpy(str, "");　　D) strcpy(str, "0");

(8) 设有如下变量定义: **int a[ ]= {1, 3, 5, 7, 9, 11, 13}, x, *p=a+3;** 则下列哪一条语句能够将数组中的**数值 5** 正确赋值给变量 **x** (B)

A) x= *p++　　B) x = * (--p)　　C) x = *(++p)　　D) x = *(p--)

(9) 下列代码段中，**合法且逻辑正确**的代码选项是(A)

A) **float** f, *fp= &f;　　B) **char** a[10]= "SampleString";

C) **char** *t; scanf("%s", t);　　D) **int** a[2][ ]={{1, 2}, {3, 4, 7}};

(10) 在 C 语言中，假定一维数组作为函数调用时的实参，则实际上在函数调用之时传递给函数的形参是(C)

A) 一维数组的所有元素值　　B) 一维数组第一个元素的值

C) 一维数组第一个元素的地址　　D) 以上都不对

四、输出代码程序块运行结果（假定所需的头文件已经包括）（15 分）

(1) 程序代码块(3')

```
void main( ){
    char array[30];
    // copy strings to the array
    strcpy(&array[0], "CH");
    strcpy(&array[2], "IBC");
    strcpy(&array[3], "NA!");
    printf("%s\n",array);
}
```

程序结果：

CHINA!

(2) 程序代码块(3')

```
void strFunc(char* pstr){

int len = strlen(pstr);

    for(int i=len-1; i>=0; i--){
        printf("%c ", *(pstr+i) );
    }
}

void main(){
char strArray[20] = "Hello,World!";
strFunc(strArray);
}
```

程序结果：

! d l r o W , o l l e H

(3) 程序代码块(3')

```
int glbCnt; /* global var */

void func(){
    static int stcVar=5; /* static var */
    int num=5; /* default storage type */

    printf("num=%d, stcVar=%d, glbCnt=%d\n",
            --num,--stcVar, glbCnt++);
}

void main(){
    // run the func for 5 times
    for(int i=0; i<5; i++){
        func();
    }
}
```

程序结果：

num=4, stcVar=4, glbCnt=0

num=4, stcVar=3, glbCnt=1

num=4, stcVar=2, glbCnt=2

num=4, stcVar=1, glbCnt=3

num=4, stcVar=0, glbCnt=4

(4) 程序代码块(3')

```
typedef struct Key{
    char *keyword;
    int keyno;
}Map;

void main(){
Map set[] = { {"design",123},{"module",456},
              {"test", 789} };

    Map *mPtr;
    int id;
    char *strPtr;
    mPtr = set; /* get the map array header */

    id = mPtr->keyno;
    strPtr = mPtr->keyword;
    printf("%s, %d\n", strPtr, id);

    id = (++mPtr)->keyno;
    strPtr = (mPtr++)->keyword;
    printf("%s, %d\n", strPtr, id);

    id = ++mPtr->keyno;
    strPtr = ++mPtr->keyword;
    printf("%s, %d\n", strPtr, id);
}
```

程序结果：

design, 123

module, 456

est, 790

(5) 程序代码块(3')

```
#define SIZE 10
int recursionFunc(int b[], int len ); /* prototype */

void main(void){
   int result;
   int array[SIZE] = {1, 2, 3, 4, 5, 6, 7, 8, 9, 10};

   result = recursionFunc(array, SIZE);
   printf("result is %d\n", result);
} /* end main */

int recursionFunc(int b[], int len ){

   if( len==1 ){ /* base case */
      return b[0];
   }
   else{ /* recursion step */
      return b[len-1] + recursionFunc(b, len-1);
   }

} /* end function recursionFunc */
```

程序结果：

result is 55

五、程序改错（20 分）

题目要求：不得改变程序框架，不得重写程序，无需文字说明，可以直接在代码上添加、删除和修改，也可以将结果填写在有“代码修正”的位置处。

(1) 从键盘获得两个整数，输出较小的整数。(2')

源代码	*代码修正*
`int x, y;`	
`scanf("%d%d", x, y);`	`scanf("%d%d", &x, &y);`
`printf("the min is %d\n, (x>y)?x:y" );`	`printf("the min is %d\n, (x<y)?x:y" );`

(2) 对 10 个元素数组进行初始化，数组元素值域为[1, 3, 5, 7, … 19]。(3')

源代码	*代码修正*
`#define SIZE=10`	`#define SIZE 10`
`void main( void ) {`	`int main(void){`
`int a[SIZE] = {0}, i;`	
`for ( i = 1, i <= SIZE, i++ )`	`for( i=1; i<=SIZE; i++ )`
`a[i] = 2 * (i-1);`	`a[i-1] = 2*i-1 ;`
`return 0;`	~~`return 0; //如果不修改 main 原定义`~~
`}`	

(3) 函数计算三个整数的平方值与立方值，需要将结果反馈给主调函数。(5')

源代码	*代码修正*
`void calculate(int x, int y, int sqr){`	`int calculate(int x, int y, int *sqr){`
`int *temp;`	`int temp;`
`sqr = x*x + y*y + z*z;`	`sqr = x*x + y*y+ z*z;`
`*temp = x*x*x + y*y*y + z*z*z;`	`temp = x*x*x + y*y*y + z*z*z;`
`return *temp;`	`return temp;`
`}`	

(4) 函数 void swap (char *, char *, int)实现两个长度相同字符串的内容交换。(4')

源代码	*代码修正*
`void swap(char *pstr1,char *pstr2,int len){`	
`char *tmp;`	`char tmp[len+1];`
`tmp = pstr1;`	`strcpy(tmp, pstr1);`
`pstr1 = pstr2;`	`strcpy(pstr1, pstr2);`
`pstr2 = tmp;`	`strcpy(pstr2, tmp);`
`}`	

(5) 将键盘输入的字符串进行大小写转换，小写字符转化为大写字符，大写字符转化为小写字符，最后将转换后的字符串进行输出。（6’）

源代码:（此题可直接将修改结果放置在卷子对应空白处）

```
#define SIZE   100 // 若此处未给出，则需要在后面声明数组是替换 SIZE
void charConversion(char array[]);
int isUpperCase(char ch);
int isLowerCase(char ch);

int main(){ // void main(){ 可以不修改

    char chArray[SIZE] = {0};   // char chArray[100]={0};

    printf("Please input a string: \n");
    scanf("%100s", chArray);


    charConversion(chArray[0]); // charConversion(chArray);

    printf("\nThe converted string: \n");
    printf("%s\n",chArray[0]); // printf("%s\n",chArray);
}

void charConversion(char array[]){

    int i=0;

    while(array[i]){

        if( isUpperCase(array[i]) ){

            array[i] = array[i] - ('a'-'A'); // '-' 改为'+'
        }

        if( isLowerCase(array[i]) ){ // 改为 else if(…)


            array[i] = array[i] + ('a'-'A'); // '+' 改为'-'
        }
         i++; //此处为增加代码

    } /* end of while */
} /* end of charConversion */

int isUpperCase(char ch){

    return (ch>'A'&&ch<'Z');// return (ch>='A'&&ch=<'Z');
}

int isLowerCase(char ch){

    return (ch>'a'&&ch<'z');// return (ch>='a'&&ch=<'z');
}
```

参考说明：每空 0.5 分，共 12 处，酌情给分。

函数调用，array，&array[0]， &array 效果一样。

六、程序填空（15’）

(1) 完善函数功能，函数描述：**计算整型数组所有元素的和。(3’)**

```
int sumArray(int array[], int size){

    int i;
    int result = 0;

    for( #1                          ){

        result =  #2                        ;
    }

     #3                      
}
```

#1: **i=0; i<size; i++**

#2: **result+array[i];**

#3: **return result;**

(2) **已知斐波那契数列**：0, 1, 1, 2, 3, 5, 8, 13, 21, …定义 fibonacci(0) = 0; fibonacci(1) = 1; fibonacci(n) = fibonacci(n – 1) + fibonacci(n – 2);**利用递归方法实现** fibonacci 函数。**(3’)**

```
int fibonacci(int n){

    if( #1          )
        return 0;
    else if( #2          )
        return 1;
    else
        return  #3                      ;
}
```

#1: **n==0**

#2: **n==1**

#3: **fibonacci(n-1) + fibonacci(n-2);**

(3) 完成程序功能，要求能够正确输出二维数组元素地址及对应值，使用**指针+偏移量**的访问方法实现。**(3’)**

```
void main(){

    int x[2][3] = {10, 20, 30, 40, 50, 60};

    int i = 0, j = 0;

    int  #1              ;

    for(i=0; i<2; i++){
        for(int j=0; j<3; j++){

            printf("[%p]:%d \t",  #2   ,  #3    );
        }

        printf("\n");
    }
}
```

#1: **(*p)[3]=x;**

#2: ***(p+i)+j**

#3: ***(*(p+i)+j)**

(4) 完善函数功能，函数描述：整型数组的冒泡排序，按照升序从小到大排序。**(3')**

```
void bubbleSort(int x[], int xSize){

     int i = 0, j = 0, temp = 0;

     for( _#1_____________ ){
          for( _#2_____________ ){
               if( _#3_____________ ){

                    temp = x[j];
                    x[j] = x[j+1];              ;
                    x[j+1] = temp;
               } // end of if
          } // end of inner for
     } // end of outer for
} // end of func
```

#1: i=1; i<xSize; i++

#2: j=0; j<xSize-i; j++

#3: x[j] > x[j+1]

(5) 完善函数功能，函数描述：利用迭代方法实现折半查找(二分查找)算法，**若没找到返回-1**。**(3')**

```
int binarySearch_iteration(int array[], int size, int key){

     int low = 0;
     int high = size-1;
     int mid;

     while( _#1_____ ){

          mid = (low+high)/2;

          if(array[mid]==key){

               _#2_____;
          }
          else if(array[mid]>key){

               high = mid-1;
          }
          else{// array[mid]<key

               _#3_____;
          }
     } // end of while control

     return -1; // no found
}
```

#1: low<=high

#2: return mid

#3: low = mid+1;

六、编写程序（25 分，6’+7’+12’）

题目要求：根据题目需要完成代码编写，不得使用全局变量，不得使用 goto 语句，字迹工整，逻辑清晰，程序结构合理。两个标准头文件 stdio.h 和 stdlib.h 勿用复写，其他头文件若使用，需要自行增加到代码中。可以使用宏定义。

(1) 编写程序：提供两个整数的输入，并计算其最大公约数和最小公倍数，并将结果输出。要求**最大公约数以函数的形式提供**。(6’)

```
int GCD(int a, int b);

int main(){

    int num1, num2, gcd;

    printf("input 2 numbers: ");
    scanf("%d%d", &num1, &num2);

    gcd = GCD(num1, num2);

    printf("%d and %d GCD is %d\n", num1, num2, gcd);
    printf("%d and %d  LCD is %d\n", num1, num2, num1*num2/gcd);
    return 0;
}

int GCD(int a, int b){

    int gcd = (a<b)? a : b;

    for(; gcd>1; gcd--){

        if( (a%gcd==0) && (b%gcd==0) )
            return gcd;
    }

    return 1; // return gcd;
}
```

参考说明：函数原型声明 1 分，函数实现 2-3 分，main 函数主体部分 2 分。

(2) 编写函数 myStringFunc 实现两个字符串交替复制功能，比如字符串 1 为"ac024"，字符串 2 为"bd135"，调用该函数后，会生成一个**新的字符串**，新字符串的内容为"abcd012345"。**参考测试样例：**字符串 1 为"ac024"，字符串 2 为"bd"，新字符串为"abcd024"；字符串 1 为"1234"，字符串 2 为"abcdef"，新字符串为"1a2b3c4def"。**myStringFunc** 要求：**1. 函数返回值：新字符串首地址；2：函数参数：**提供两个字符串作为参数，要求函数对该字符串参数不能修改。

请给出 **myStringFunc 原型声明**以及**具体实现**。(7')

```
char* myStringFunc(char *s1, char *s2); // prototype


char* myStringFunc(char *s1, char *s2){

    // allocate the new string memory
    char* strResult = (char *)malloc( strlen(s1)+strlen(s2)+1 );

    // get the header of the memory
    char* strPtr = strResult;

    // while not reach the end of the strings, need continue
    while(*s1||*s2){

        if(*s1){//if the str1 not end, save it in the new string

            *strResult = *s1;
            s1++;
            strResult++;
        }

        if(*s2){//if the str2 not end, save it in the new string

            *strResult = *s2;
            s2++;
            strResult++;
        }
    }

    *strResult = '\0';  // end of the string tag
    return strPtr;   // return the header of the allocated
memory
}
```

参考说明：函数原型声明 1 分，函数逻辑结构正确 2-3 分，红色关键部分 3 分，其他酌情给分。

(3) 一个公司有若干名员工，每名员工有姓名，性别，工龄，工资，工资发放币种组合等信息，现需要完成一个简单的员工信息与工资管理小程序，要求如下：

1. 定义员工**数据结构** Worker，用于记录以上员工档案信息，便于工资发放的各种钞票数（工资为整数，发放的工资各种钞票限定为 100 元，50 元，20 元，10 元，5 元，1 元，发放的钞票数张数要求为最少）；

2. 能够根据工资从高到低对员工档案信息进行排序；

3. 最后输出工龄大于 10 年，工资高于 5000 元的所有女员工信息。

程序要求如下：

1. 提供 void getWorkerInfo(Worker* worker)**函数**，记录**一个员工的信息**，并通过 worker 返回给主调函数。(3')

2. 提供 void sortWorkersByWage(Worker* workers, int len)函数,对员工信息结构体数组进行排序，排序条件为**员工工资条件（由高到低）**。(3')

3. 提供 void displaySingleWorker(Worker worker)函数，能够对一个员工的信息进行完整的显示，显示顺序为姓名、性别、工龄、工资、工资发放组合，一个具体示例如下：Tom,M,10,8585,[100:85 50:1 20:1 10:1 5:1 1:0]。(2')

4. 提供 void displayWorkers(Worker* workers, int len)函数，通过 displaySingleWorker 对员工信息结构体数组进行统一输出显示。(2')

5. 编写主程序，实现并使用定义的函数，要求能够读入 5 个员工信息，并对员工信息根据工资进行排序,输出排序后员工的信息,最后输出满足条件的女员工信息。(12')

```
#define SIZE       5
#define LENGTH     80
#define GENDER_M    'M'
#define GENDER_F    'F'

typedef struct worker{
    char name[LENGTH];
    char gender;    // M:male, F:female
    int workAge;
    int wage;
    int combi[6]; //[0]:100,[1]:50,[2]:20,[3]:10,[4]:5,[5]:1,
money counter
}Worker;

void getWorkerInfo(Worker* worker);
void sortWorkersByWage(Worker* workers, int len);
void displaySingleWorker(Worker worker);
void displayWorkers(Worker* workers, int len);
void countMoneyComi(int* array, int len, int wage);

int main(){

    Worker workers[SIZE];
    int i;

    for(i=0; i<SIZE; i++){
        getWorkerInfo(&workers[i]);
    }

    printf("Original workers list: \n");
    printf("%-15s%-4s%-9s%-5s%-12s\n",
            "name","sex","work_age","wage","moneyCombi");
    displayWorkers(workers, SIZE);
    // sort the workers
    sortWorkersByWage(workers, SIZE);
```

```
    printf("\nSorted workers list: \n");
    printf("%-15s%-4s%-9s%-5s%-12s\n",
         "name","sex","work_age","wage","moneyCombi");
    displayWorkers(workers, SIZE);

    // display the found workers information
    printf("\nFound workers: \n");
    printf("%-15s%-4s%-9s%-5s%-12s\n",
            "name","sex","work_age","wage","moneyCombi");
    for(i=0; i<SIZE; i++){

        if(workers[i].workAge>10 && workers[i].wage>5000
           && workers[i].gender == GENDER_F ){
           displaySingleWorker(workers[i]);
        }
    }
    return 0;
}

void getWorkerInfo(Worker* worker){

    int moneyTmp = 0;

    printf("worker name: ");
    scanf("%s", worker->name);  // *worker.name
    setbuf(stdin, NULL);

    printf("gender(M-Male, F-female): ");
    scanf("%c", &(worker->gender));

    printf("work year: ");
    scanf("%d", &(worker->workAge));
    setbuf(stdin, NULL);

    printf("worker wage: ");
    scanf("%d", &(worker->wage));

    // count the money cominations
    countMoneyComi(worker->combi, 6, worker->wage);

    printf("\n");
}

void sortWorkersByWage(Worker* workers, int len){

    for(int round=1; round<len; round++){
        for(int i=0; i<len-1; i++){
            if(workers[i].wage < workers[i+1].wage){
                // swap the two struct workers
                Worker tmp;
                tmp = workers[i];
                workers[i] = workers[i+1];
                workers[i+1] = tmp;
            }
        }// end of inner for
    }// end of outer for
}

void displaySingleWorker(Worker worker){

    int money[] = {100, 50, 20, 10, 5, 1};

    printf("%-15s%-4c%-9d%-5d", worker.name,
           worker.gender, worker.workAge, worker.wage);
    printf("[ ");
```

```
for(int i=0; i<6; i++){
        printf("%2d:%3d ", money[i], worker.combi[i]);
    }
    printf("]\n");
}

void displayWorkers(Worker* workers, int len){

    for(int i=0; i<len; i++){
        displaySingleWorker(workers[i]);
    }
}

void countMoneyComi(int* array, int len, int wage){

    array[0] = wage/100;
    wage = wage%100;

    array[1] = wage/50;
    wage = wage%50;

    array[2] = wage/20;
    wage = wage%20;

    array[3] = wage/10;
    wage = wage%10;

    array[4] = wage/5;
    array[5] = wage%5;
}
```

参考说明：函数原型声明 1 分，每个函数实现 2-3 分，其他酌情给分，可以定义辅助函数简化代码。

附录：

- **ASCII 字符表**

ASCII character set

	0	1	2	3	4	5	6	7	8	9
0	nul	soh	stx	etx	eot	enq	ack	bel	bs	ht
1	lf	vt	ff	cr	so	si	dle	dc1	dc2	dc3
2	dc4	nak	syn	etb	can	em	sub	esc	fs	gs
3	rs	us	sp	!	"	#	$	%	&	‘
4	(	)	*	+	,	-	.	/	0	1
5	2	3	4	5	6	7	8	9	:	;
6	<	=	>	?	@	A	B	C	D	E
7	F	G	H	I	J	K	L	M	N	O
8	P	Q	R	S	T	U	V	W	X	Y
9	Z	[	\	]	^	_	’	a	b	c
10	d	e	f	g	h	i	j	k	l	m
11	n	o	p	q	r	s	t	u	v	w
12	x	y	z	{	|	}	~	del		

Fig. B.1 | ASCII Character Set.

The digits at the left of the table are the left digits of the decimal equivalent (0–127) of the character code, and the digits at the top of the table are the right digits of the character code. For example, the character code for “F” is 70, and the character code for “&” is 38.

- **常用函数表**

函数原型	**函数描述**
char *strcpy(char *s1, const char *s2)	
描述：	拷贝字符串 s2 内容到字符数组 s1 中。返回值为 s1。
char *strcat(char *s1, const char *s2)	
描述：	附加字符串 s2 内容到字符数组 s1 中。返回值为 s1。
int strcmp(const char *s1, const char *s2);	
描述：	比较字符串 s1 和字符串 s2。 函数返回值 0，小于 0 或者大于 0，分别代表 s1 等于，小于或者大于 s2。
void srand(unsigned int seed);	
描述	随机数发生器的初始化函数。
int rand(void);	
描述：	该函数返回一个在 0 和 RAND_MAX 之间的伪随机数。
void *malloc(size_t size);	
描述：	该函数指向大小为 size 内存的指针，如果失败返回 NULL。
void free(void* ptr);	
描述：	该函数释放指针所指向的内存区域。