Introduction to Computer Science and C Programming-Quiz2

2016/12/12

1. (32 pts) **Multiple Selection Questions** *(There may exist one or more than one answer. You will lose ? pts till there is no pts to lose for one wrong choice in each question)*

(1) Which of the following declarations are able to show **exactly "2016" string** and won't cause a program crash when using *puts(s)* statement? BCDE

ANS: _____

(A) char *s; s[0] = '2'; s[1] = '0'; s[2] = '1'; s[3] = '6'; s[4] = '\0'

(B) char *s = "2016"

(C) char s[10] = "2016";

(D) char s[5] = "2016";

(E) char s[] = "2016";

Details:

(A) The pointer s hasn't been pointed to any memory, it's illegal to assign values to unknown memories.

(2) Which of the following statements correctly statically/dynamically allocate an array of 100 values of integer type, and stores a pointer to that array in *ptr*? BD

ANS: _____

(A) int *ptr = (int *) malloc(100);

(B) int *ptr = (int *) malloc(100 * sizeof(int));

(C) int *ptr = (int *) calloc(100 * sizeof(int));

(D) int ptr[100];

Details:

(A) The size in malloc should be "total bytes" that you would like to allocate. By default (32-bit windows C compiler), this statement will allocate only 25 integers.

(C) It should be **int *ptr = (int *) calloc(100, sizeof(int))**;

(3) Which of the following statements are **true**? CDEF

ANS: _____

(A) A structure and a union with the same members always occupy the same size of memory.

(B) The ***malloc*** function can allocate space for an array, and it initializes the memory that it allocates

(C) If ***realloc(ptr, size)*** is called with 0 as its second argument, it frees the memory block.

(D) To find the address of a variable, we use the '&'operator; to gain access to the object that a pointer points to, we use the '*' operator.

(E) If function arguments are passed by value, means any modification to function parameters won't affect the corresponding arguments

(F) *int f (int a[]) {…}* is a valid function definition. (… represent any valid statements)

Details:

(A) The size of a structure is the sum of members; however, a union's size is the largest member's size.

(B) ***malloc*** won't initialize the memory, ***calloc*** is the one that will.

(4) Which of the following statements about arrays and linked lists are **true**? ACE

ANS: _____

(A) After declaring a static array and the size of it, you can't resize it in the runtime.

(B) You are able to randomly access an element in a linked list.

(C) An array occupies continuous memory space, while a linked list may not..

(D) Accessing an element in an array is fast if the element is close to the beginning of the array, slow if it's near the end.

(E) If both an array and a linked list are arranged in descending order, it's faster to find the k_{th} biggest element in an array than in a linked list

Details:

(B) To access an element in a linked list, you should always traverse from the first of it.

(D) Indexing an array takes the same time.

2. (18 pts) Please write the output of following program.

```c
#include <stdio.h>

void copy0(int a, int b){
    a = b;
}
void copy1(int *a, int b){
    *a = b;
}
void copy2(int a, int *b){
    a = *b;
}
void copy3(int *a, int b){
    a = &b;
}
void copy4(int *a, int *b){
    a = b;
}
void copy5(int *a, int *b){
    *a = *b;
}

int main(){
    int a[6] = {-1};
    int b[] = {0, 1, 2, 3, 4, 5};

    copy0(a[0], b[0]);
    copy1(&a[1], b[1]);
    copy2(a[2], &b[2]);
    copy3(&a[3], b[3]);
    copy4(&a[4], &b[4]);
    copy5(&a[5], &b[5]);

    int i;
    for(i = 0; i < 6; i++){
        printf("a[%d] = %d\n", i, a[i]);
    }

    return 0;
}
```

ANS:

a[0] = -1

a[1] = 1

a[2] = 0

a[3] = 0

a[4] = 0

a[5] = 5

3. (28 pts) Complete the following program.

```c
#include <stdio.h>
void fun1(int *p1, int *p2, int *p){
    if(*p1 <= *p2){
        *p = *p1;
    }else{
        *p = *p2;
    }
}

int *fun2(int *p1, int *p2){
    if(*p1 >= *p2){
        return p1;
    } else{
        return p2;
    }
}

int main(){
    int a = 10, b = 20, *c, d;
    c = &d;
    fun1(&a, &b, c);
    printf("min(%d %d) = %d\n", a, b, d);

    c = fun2(&a, &b);
    printf("Max(%d, %d) = %d\n", a, b, *c);

    return 0;
}
```

<<Program Output>>

min(10, 20) = 10
Max(10, 20) = 20

4. (22 pts) Please write the output of following program.

(1)(8 pts)

```
int a[]={'a','d','g','j','m','p','s','v','y'};
int *q=a;
printf("%c\n",*q+8);     //----(A)
printf("%c\n",(*++q));   //----(B)
printf("%c\n",*(q+=4));  //----(C)
printf("%c\n",*(q+1));   //----(D)
```

(2)(9 pts)

```
int b[15]={2,4,6,8,10,12,14,16,18,20,22,24,26,28,30};
int c[3][5]={2,4,6,8,10,12,14,16,18,20,22,24,26,28,30};
int *p=b;
printf("%d\n",*(p+4));                 //----(A)
printf("%d\n",*(c+2)[0]);              //----(B)
printf("%d\n",(*(c+0)[1])**(p+5)*6); //----(C)
```

(3)(5 pts)

```
#include <stdio.h>
void fun(int a[],int l,int m,int h){
    int b[10],i;
    for (i=m+1;i<=h;i++)  b[i-m-1]=a[i];
    i=m;
    int j=h-m-1,k=h;
    while (i>=0&&k>i){
        if (a[i]>b[j]) a[k--]=a[i--];
        else a[k--]=b[j--];
    }
    while (k>i)
        a[k--]=b[j--];
}
int main(){
int b[13]={7,8,9,10,11,12,13,1,2,3,4,5,6};
    fun(b,0,6,12);
    for (i=0;i<13;i++)  printf("%d ",b[i]);
    printf("\n");
}
```

ANS:

(1)

(A)i

(B)d

(C)p

(D)s

(2)

(A)10

(B)22

(C)864

(3)

1 2 3 4 5 6 7 8 9 10 11 12 13