

# 2021 程序设计b卷答案

## 一、单选题

1. D  
main函数可以被其他函数及其自己调用,也可以添加命令行参数
2. A  
B不能数字开头,C保留关键字,D .在标识符中是非法字符
3. B
4. D
5. D
6. B
7. B  
执行case10和case11中的内容
8. D
9. B
10. D(其实空格有问题)
11. B  
p[1\*3+0]也就是a[1][0]也就是第四个元素
12. B  
表示内容为r->id;
13. D  
(\*++p)[1]即a[1][1],(\*p)[1]即p[0][1]
14. D  
\*p2是char类型,参数不对
15. B  
x末尾有'\0',y无
16. D
17. C
18. B  
A是数组类型,C不合法,D是函数类型
19. D
20. A  
ps是指针,在64位程序中大小为8字节;\*ps是char类型;ps数组有10+1('\0')个元素;buffer字符串有5个元素;数组大小为6('\0')
21. D  
NULL表示不指向任意内容,具体根据编译器实现,一般为0x0,
22. A
23. C  
++最后执行
24. D

## 二、多选题

1. AD
2. BD
3. ABD  
预处理命令可以在其他位置(似乎是任意位置,例如#ifdef),预处理命令由预处理器处理,不由编译器(核心)处理,不属于c语言本身.
4. BC  
D a大小是11,该种初始化方式会添加'\0'
5. BCD
6. ABD  
C语言的早期版本只允许在函数开头定义局部变量,因此可以认为c正确,(现在版本允许在函数内的任意大括号内定义局部变量,只在该大括号内可用)

## 三、填空

1. 213.83
2. 1  
\X41表示用16进制表示的ascii码
3. 11  
10个字符加'\0'
4. 0  
\*p++是取p指向内容,为'0',然后把p自增,'0'值是48,不是0
5. 200 100
6.  

```
((!(x>='a')&&(x<='z')>))&&((y>='a')&&y<='z'))  
||(((x>='A')&&(x<='Z'))&&((y>='A')&&(y<='Z'))))
```
7. 0
8. -1
9. 0  
表达式等价于a[2][0],未被初始化赋值的部分

## 四、程序填空题

```
1. (1) (a[i]%j)!=0  
(2) continue  
(3) j==a[i]  
(4) a  
2. (5) j=0  
(6) str  
(7) str[i]!='\0'  
(8) ((str[i]>='a')&&(str[i]<='z'))||  
    ((str[i]>='A')&&(str[i]<='Z'))  
3. (9) N+1  
(10) m  
(11) i-1  
(12) m  
4. (13) salary/500  
(14) return salary*0.05;  
(15) return salary*0.15;  
5. (16) mid+1  
(17) mid  
(18) mid  
(19) -1
```

## 编程题

1.  

```
int find_str(char *str1, char *str2)
{
    int i, j, flag = -1;
    for (i = 0, j = 0; str1[i] != NULL; i++)
    {
        while (str1[i] == str2[j] && str1[i] && str2[j])
        {
            i++;
            j++;
            if (str2[j] == NULL)
            {
                flag = i - j;
                return flag;
            }
            j = 0;
        }
        return flag;
    }
}
```
2.  

```
#include <stdio.h>
#define N 5
struct employee
{
    char name[30];
    int id;
    float salary;
} empy[N];
void indata(struct employee *empy)
{
    for (int i = 0; i < N; i++)
    {
        scanf("%s%d%f", empy[i].name, &empy[i].id, &empy[i].salary);
    }
}
void sort(struct employee *empy)
{
    int i, j;
    struct employee tmp;
    for (i = 0; i < N - 1; i++)
        for (j = 0; j < N - 1 - i; j++)
            if (empy[j].id > empy[j + 1].id)
            {
                tmp = empy[j];
                empy[j] = empy[j + 1];
                empy[j + 1] = tmp;
            }
}
int main()
{
    indata(empy);
    sort(empy);
    findata(empy);
    foutdata(empy);
    return 0;
}
```
3.  

```
#include <stdio.h>
#include <stdlib.h>
typedef struct Link
{
    char c;
    struct Link *next;
} link;

link *initlink(int num)
{
    link *head = (link *)malloc(sizeof(link));
    head->next=NULL;
    link *tmp = head;
    for (int i = 0; i < num; i++)
    {
        tmp=(link*)malloc(sizeof(link));
        tmp->next = head->next;
        head->next=tmp;
        scanf("%c", &(tmp->c));
    }
    return head;
}
link *reverselink(link *L)
{
    link *p;
    p = L->next;
    L->next = NULL;
    while (p != NULL)
    {
        link *tmp;
        tmp = p->next;
        p->next = L->next;
        L->next = p;
        p = tmp;
    }
    return L;
}
int main()
{
    link *h = initlink(5);
    h = reverselink(h);
    while (h->next != NULL)
    {
        h = h->next;
        printf("%c", h->c);
    }
    return 0;
}
```