哈尔滨工业大学（深圳）2023 年秋季学期

高级语言程序设计试题（A）（回忆版）

试题回忆：CLERI23 小组

alphabet, doctxing, firework, flaneur, Moon, Mourinho, sans, sleepfish, who, 源, Zhexi, Zjl37

本卷为闭卷考试，满分 100 分，考试时间 120 分钟。

一、单项选择题（本题共 25 小题，每题 2 分，共 50 分）

1. 以下错误的描述是

A. 使用 while 和 do-while 循环时，循环变量初始化的操作应在循环语句之前完成

B. while 循环是先判断表达式，后执行循环语句

C. do-while 循环和 for 循环均是先执行循环体语句，后判断表达式

D. for，while，do-while 循环中的循环体均可由空语句构成

2. 以下代码的运行结果是

```c
#include <stdio.h>
main()
{
    char c = 'A';
    if('0' <= c <= '9')
        printf("YES");
    else
        printf("NO");
}
```

A. `YES`

B. `NO`

C. `YESNO`

D. 语句错误

3. 下面初始化正确的是

A. `char str[8] = { "language" }`

B. `int a[5] = {0, 1, 2, 3, 4, 5}`

C. `char* str = "abcd"`

D. `int a[2][] = {1, 2, 3, 4, 5, 6}`

4. 下面定义了一个共用体，最有可能是`sizeof(test)`的结果的是

```
union
{
    char ch;
    int a;
    float b;
} test;
```

A. 4

B. 5

C. 9

D. 12

5. 当输入 9 时，以下程序运行结果是

```
#include <stdio.h>
main()
{
    int n;
    scanf("%d", &n);
    if (n++ < 10)
        printf("%d\n", n);
    else
        printf("%d\n", n--);
}
```

A. 11

B. 10

C. 9

D. 8

6. 以下程序运行结果是

```
#include <stdio.h>
void fun(int *x, int *y)
{
    printf("%d %d", *x, *y);
    *x = 3;
    *y = 4;
}

main()
{
    int x = 1, y = 2;
    fun(&y, &x);
    printf("%d %d", x, y);
)
```

A. `2 1 4 3`

B. `1 2 1 2`

C. `1 2 3 4`

D. `2 1 1 2`

7. 下面程序的输出是

```c
#include <stdio.h>
int main()
{
    char a[10] = { 9, 8, 7, 6, 5, 4, 3, 2, 1, 0 }, *p = a + 5;
    printf("%d", *--p);
    return 0;
}
```

A. 无法编译

B. `a[4]`的地址

C. `5`

D. `3`

8. 如果`short int`所占字节数为 2，则`unsigned short int`可存储的数据取值范围为

A. 0 ~ 255

B. 0 ~ 65535

C. −128 ~ 127

D. −32768 ~ 32767

9. 以下程序的运行结果是

```c
#include <stdio.h>

int f(int m)
{
    int a, b = 0;
    static int c = 3;
    a = c++, b++;
    return (a);
}
int main()
{
    int a = 2, i, k;
    for (i = 0; i < 2; i++)
    {
        k = f(a++);
    }
    printf("%d\n", k);
    return 0;
}
```

A. `0`

B. `3`

C. `4`

D. `5`

10. 以下程序的运行结果是：

```c
#include <stdio.h>
int Fun(int m)
{
    static int n = 1;
    m /= 2;
    m = m * 2;
    if (m)
    {
        n *= m;
        return (Fun(m - 2));
    }
    else
        return n;
}
int main()
{
    int a, i;
    for (i = 0; i < 2; i++)
    {
        a = Fun(4 + i);
        printf("%d\n", a);
    }
    return 0;
}
```

A. `8 8`

B. `0 0`

C. `64 64`

D. `4 4`

11. 以下程序的输出是

```c
#include <stdio.h>
long fun(int n)
{
    long s;
    if (n == 1 || n == 2)
        s = 2;
    else
        s = n - fun(n - 1);
    return s;
}

main()
{
    printf("%ld", fun(3));
    return 0;
}
```

A. 1

B. 2

C. 3

D. 4

12. 以下程序的输出是

```
#include <stdio.h>
#include <stdlib.h>
amovep(int *p, int (*a)[3], int n)
{
    int i, j;
    for (i = 0; i < n; i++)
    {
        for (j = 0; j < 3; j++)
        {
            *p = a[i][j];
            p++;
        }
    }
}

main()
{
    int *p, a[3][3] = {{1, 3, 5}, {2, 4, 6}};
    p = (int *)malloc(100);
    amovep(p, a, 3);
    printf("%d %d", p[2], p[5]);
    free(p);
}
```

A. 5 6

B. 2 5

C. 3 4

D. 程序错误

13. 以下程序运行后，文件 test 最终的内容是

```
#include <stdio.h>
#include <string.h>
void fun(char *fname, char *st)
{
    FILE *fp = fopen(fname, "w");
    for (int i = 0; i < strlen(st); i++)
        putc(st[i], fp);
}
int main()
{
    fun("test", "new world");
    fun("test", "hello");
    return 0;
}
```

A. `new world`

B. `hello`

C. `new worldhello`

D. `hello rld`

14. 以下程序的输出是

```
#include <stdio.h>
void fun(char *c, char d)
{
    *c = *c + 1, d = d + 1;
    printf("%c,%c,", *c, d);
}
main()
{
    char a = 'A', b = 'a';
    fun(&b, a);
    printf("%c,%c\n", a, b);
}
```

A. `A,b,A,a`

B. `B,b,A,a`

C. `A,b,B,a`

D. `b,B,A,b`

15. 以下程序的输出是

```
#include <stdio.h>
#include <stdlib.h>
struct Node
{
    int num;
    struct Node *next;
};
main()
{
    struct Node *p, *q, *r;
    p = (struct Node *)malloc(sizeof(struct Node));
    q = (struct Node *)malloc(sizeof(struct Node));
    r = (struct Node *)malloc(sizeof(struct Node));
    p->num = 10;
    q->num = 20;
    r->num = 30;
    p->next = q;
    q->next = r;
    printf("%d", p->num + q->next->num);
}
```

A. 10

B. 20

C. 30

D. 40

16. 下列说法正确的是

A. 注释可以在程序任意合适的地方出现
B. 花括号只能用于界定函数体
C. 程序的基本结构是函数，用户可以定义任何名称的函数
D. 分号只用于分隔语句，不是语句的一部分

17. 以下错误的描述是

A. 不同的函数中可以使用相同的变量名
B. 形式参数是局部变量
C. 函数内部定义的变量只有在本函数的范围内有效
D. 在函数内部的复合语句中定义的变量可以在本函数的范围内有效

18. 有以下程序

```c
#include <stdio.h>
struct s
{
    int x, y;
} data[2] = {10, 100, 20, 200};

main()
{
    struct s *p = data;
    printf("%d", ++(p->x));
}
```

程序运行后的输出结果是

A. 10　　B. 11
C. 20　　D. 21

19. 以下函数的作用是

```c
fun(char *p2, char *p1)
{
    while((*p2 = *p1) != '\0')
    {
        p1++;
        p2++;
    }
}
```

A. 将 p1 所指的字符串复制到 p2 所指的内存空间
B. 将 p1 所指的字符串地址赋值给 p2
C. 比较 p1 与 p2 所指的字符串大小
D. 检查 p1，p2 两指针所指字符串中是否含有 `'\0'`

20. 已知有 `int a, *pa = &a`，能正确为 `a` 读入数据的是

A. `scanf("%d", pa);`

B. `scanf("%d", a);`

C. `scanf("%d", &pa);`

D. `scanf("%d", *pa);`

21. 以下程序的输出是

```
#include <stdio.h>
void swap1(int c[])
{
    int t;
    t = c[0]; c[0] = c[1]; c[1] = t;
}

void swap2(int c0, int c1)
{
    int t;
    t = c0; c0 = c1; c1 = t;
}

main()
{
    int a[2] = {3, 5}, b[2] = {3, 5};
    swap1(a);
    swap2(b[0], b[1]);
    printf("%d %d %d %d", a[0], a[1], b[0], b[1]);
}
```

A. `5 3 5 3`

B. `5 3 3 5`

C. `3 5 3 5`

D. `3 5 5 3`

22. 以下程序的输出是

```
#include <stdio.h>
#include <string.h>
main()
{
    char *p[10] = { "abc", "aabdfg", "dcabe", "abbd", "cd" };
    printf("%d", strlen(p[4]));
}
```

A. 2

B. 3

C. 4

D. 5

23. 以下程序的输出是

```
#include <stdio.h>
int main()
{
    int i;
    for (i = 0; i < 3; i++)
    {
        switch (i)
        {
            case 0: printf("%d", i);
            case 2: printf("%d", i);
            default: printf("%d", i);
        }
    }
}
```

A. `022111`

B. `021021`

C. `000122`

D. `012`

24. 若变量都已正确定义，要求计算 5!，下列选项不能完成的是

A. `for (i = 1, p = 1; i <= 5; i++) p *= i;`

B. `for (i = 1; i <= 5; i++) { p = 1; p *= i; }`

C. `i = 1, p = 1; while(i <= 5) { p *= i; i++; }`

D. `i = 1, p = 1; do { p *= i; i++; } while (i <= 5)`

25. 用 `(int*)calloc(m*n, sizeof(int));` 申请了一块动态内存，并用指针变量 `p` 指向它，来保存 m*n 个整形变量，则下列通过 `p` 访问动态二维数组 i 行 j 列的正确选项是

A. `p[i*n + j]`

B. `*(*(p + i) + j)`

C. `p[j*n + i]`

D. `p[i][j]`

二、程序填空题（本题共 5 小题，每空 2 分，共 36 分）

1. 输入学生数量和他们的成绩，计算平均成绩。注：用动态数组。

```
#include <stdio.h>
#include <stdlib.h>

main()
{
    int *p, i, n, sum;
    printf("size of array:");
    scanf("%d", ____(1)____);

    p = malloc(n * sizeof(int));
    if(p == NULL)
    {
        printf("no enough memory!\n");
        ____(2)____;
    }

    for (i = 0; i < n; i++)
    {
        scanf("%d", p + i);
    }

    ____(3)____;

    for(i = 0; i < n; i++)
    {
        ____(4)____;
    }
    printf("average=%d\n", sum/n);
    free(p);
}
```

2. 以下函数计算字符串 substr 在 str 中的出现次数。

```
int check(char *str, char *substr)
{
    int x, y, z, num = 0;
    for (x = 0; str[x] != ____(1)____; x++)
    {
        for (y = ____(2)____, z = 0; str[y] == substr[z]; y++, z++)
        {
            if (substr[____(3)____] == '\0')
            {
                num++;
            }
        }
    }
    return ____(4)____;
}
```

3. 输入名字，查询其名次和分数，要求多次查询，以输入 0 为结束标志。

```
#include <stdio.h>
#include <string.h>
struct student
{
    int rank;
    char *name;
    double score;
};

____(1)____ stu[] = {
    {3, "liwang", 89.3},
    {4, "zhangyi", 78.2},
    {1, "wangqi", 95.1},
    {2, "gongwei", 90.6}
};
#define NUM 4
int main()
{
    char str[10];
    int i;
    do
    {
        printf("Please enter the name:");
        scanf("%s", str);
        for (i = 0; i < NUM; i++)
        {
            if (____(2)____)
            {
                printf("Name=%s\n", stu[i].name);
                printf("Rank=%d\n", stu[i].rank);
                printf("Score=%5.1f\n", stu[i].score);
                ____(3)____;
            }
        }
        if (i == NUM)
        {
            printf("Not Found\n");
        }
    } while (strcmp(str, "0") != 0);
}
```

4. 输入 10 个数，输出 10 个数中的最大值及其位置。

```
#include <stdio.h>

main()
{
    int a[10], pos, max, n;
    for (n = 0; n < 10; n++)
        scanf(____(1)____);
    max = ____(2)____;
    pos = ____(3)____;
    for (n = 0; n < 10; n++)
    {
        if (a[n] > max)
        {
            max = ____(4)____;
            pos = n;
        }
    }
    printf("%d %d\n", max, pos);
}
```

5. 下面的程序让用户输入字符，并且判断该字符是否为大小写字母、数字、空格或其他：

```
#include <stdio.h>
int main()
{
    char ch;
    ch = getchar();
    if (____(1)____)
    {
        puts("It is an English character.");
    }
    else if (____(2)____)
    {
        puts("It is a digit .");
    }
    else if (____(3)____)
    {
        puts("It is a space.");
    }
    else
    {
        puts("It is other character.");
    }
    return 0;
}
```

三、编程题（14 分）

利用矩阵相乘公式，编程计算 M*N 阶矩阵和 N*M 阶矩阵之积（M*M 阶矩阵）。要求：

1. 输入输出格式，形如：

```
Input 3*4 matrix a:
a[0,0]: 11
a[0,1]: 2
……

Input 4*3 matrix b:
b[0,0]: 23
b[0,1]: 21
……
b[3,1]: 7
b[3,2]: 54

Results:
  1643  2063  1063
  4144  5205  5394
  1014  1106  1923
```

2. 用宏定义分别定义 M，N 为 3，4.

3. 函数原型分别定义成：

```
void MultiplyMatrix(int a[M][N], int b[N][M], int c[M][M]);
void InputMatrixA(int a[M][N]);
void InputMatrixB(int b[N][M]);
void OutputMatrix(int a[M][M]);
```

4. 主函数定义成：

```
int main()
{
    int a[M][N], b[N][M], c[M][M];
    InputMatrixA(a);
    InputMatrixB(b);
    MultiplyMatrix(a, b, c);
    OutputMatrix(c);
    return 0;
}
```