Предусловие: во всех примерах по умолчанию подключены <stdio.h>, <malloc.h>, <string.h>, <math.h>. Машинное слово: 4 байта.

Если в исходном коде допущены ошибки, это сделано намеренно.

1. В результате выполнения программы:

```
int main() {
    unsigned int x = 4;
    if (x > -3)
        printf("1");
    else if (x < 4)
        printf("2");
    else
        printf("3");
    return 0;
}
```

на экран будет выведено значение: _____.

2. В результате выполнения программы:

```
int main() {
    float a = 2.55;
    unsigned char b = a + 2;
    printf("%d", b);
    return 42;
}
```

на экран будет выведено значение: _____.

3. В результате выполнения программы:

```
int main() {
    unsigned char b = -1;
    printf("%d", b);
    return 0;
}
```

на экран будет выведено значение: _____.

4. В результате выполнения программы:

```
int main() {
    char x = 64;
    char y = x * 2;
    int z = x * 2;
    printf("%d %d %d", x*2, y+2, z);
    return 0;
}
```

на экран будут выведены значения: _____.

5. В результате выполнения программы:

```
int main() {
    unsigned char x = 4;
    if (x < -3)
        printf("1");
    else if (x = -3)
        printf("2");
    else
        printf("3");
    return 0;
}
```

на экран будет выведено значение: _____.

6. В результате выполнения программы:

```
int main() {
    int x = sqrt(10);
    printf("%d", x);
    return 0;
}
```

на экран будет выведено значение: _____.

7. В результате выполнения программы:

```
int main() {
    int x = 0, y = 0;
    printf("%d %d ", x++, ++y);
    printf("%d %d ", ++x, y++);
    return 0;
}
```

на экран будут выведены значения: _____.

8. В результате выполнения программы:

```
int main() {
    int x = 3;
    printf("%d", (x+x)++);
    return 0;
}
```

на экран будет выведено значение: _____.

9. В результате выполнения программы:

```
int main() {
    int x = 3;
    int y = 1;
    printf("%d", (x ^ y) << 5);
    return 0;
}
```

на экран будет выведено значение: _____.

10. В результате выполнения программы:

```
int main() {
    int x = 3;
    int y = 6;
    printf("%d", x && y);
    return 0;
}
```

на экран будет выведено значение: _____.

11. Вставьте строчку в исходный код, чтобы обнулить последний байт:

```
int main() {
    int x;
    scanf("%d", &x);
    x = /* место для кода */
    printf("%d", x);
    return 0;
}
```

12. Вставьте строчку кода, содержащую тернарный оператор для поиска максимума из чисел a и b:

```
int main() {
    int a, b;
    scanf("%d", &a);
    scanf("%d", &b);
    int max = /* место для кода */
    printf("%d", max);
    return 0;
}
```

13. В результате выполнения программы:

```
int main() {
    int n = 5;
    int res = 0;
    int y = 10;
    if (n > 0)
         if (n > 10)
            res = 100;
    else if (n < 10)
        res = 3;
    printf("%d", res+y);
    return 0;
}
```

на экран будет выведено значение: _____.

14. В результате выполнения программы:

```
int main() {
    int x = 0;
    int y = 0;
    if (++x || ++y) {
        printf("%d, %d", x, y);
    }
    return 0;
}
```

на экран будут выведены значения: _____.

15. В результате выполнения программы:

```
int main() {
    int res;
    int a = 2;
    switch (a) {
        case 1:
            res = 1;
        case 2:
            res = 2;
        case 3:
            res = 3;
        case 4:
            res = 4;
    }
    printf("%d", res);
    return 0;
}
```

на экран будет выведено значение: _____.

16. В результате выполнения программы:

```
int main() {
    int a = 2;
    int res;
    switch (a) {
        case 2:
        case 1:
            res = 5;
            break;
        default:
            res = 10;
    }
    printf("%d", res);
    return 0;
}
```

на экран будет выведено: _____.

17. Замените данный фрагмент:

```
for (вып1; вып2; вып3)
    инструкция
```

«эквивалентным» через цикл while.

18. Опишите случай, когда две конструкции в задании 17 работают по-разному.

19. Какими тремя способами может быть прерван цикл

```
for (;;) {
}
```

20. В чём заключается разница между циклами do-while в C и repeat-until в Pascal.

21. Инструкции break и goto. Когда имеет смысл их применять.

22. Что будет выведено в результате работы программы:

```
int f() {
    static int x;
    if ((x = 5 * ++x) < 10)
        return 5.47 + x;
    else
        return 7*x;
}

int main() {
    printf("%d", f());
    return 0;
}
```

23. Что будет выведено в результате работы программы:

```
void f() {
    static int x = 10;
    printf("%d\n", ++x);
}

int g() {
    int x = 20;
    printf("%d\n", x++);
}


int main() {
    f();
    f();
    g();
    g();
}
```

24. Предположим, что функция main вызывала функцию f дважды.

1. Какого объема утечка памяти произошла между первым и вторым вызовом?
2. Какой объем динамической памяти был затрачен за счёт двух вызовов f?

```
int f() {
     const int N = 10;
     static int* x = (int*)malloc(sizeof(int) * N);
     // ...
}
```

25. Зачем применять ключевое слово static к функциям?

26. В результате работы программы

```
int main() {
    int i = 10;
    int s = 0;
    for (int i = 0; i < 5; i++) {
        s += i;
    }
    printf("%d, %d", i, s);
    return 0;
}
```

на экран будут выведены значения: _____.

27. Строка s была проинициализирована при объявлении:

```
int main() {
    char s[] = "Hello";
    return -10;
}
```

Для строки s будет выделена память объема ___ байт.

28. Определение «препроцессор». Директивы препроцессора #define и #include; как работают. При использовании #include в чём заключается разница между <> и "".

29. В результате работы программы:

```
int main() {
    int x = 1, y = 2;
    int *ip = &x;
    y = *ip;
    *ip = 0;
    printf("%d, %d, %d", x, y, *ip);
    return 0;
}
```

будет выведены значения _____.

30. В результате работы программы:

```
int main() {
    int z[] = {1, 0, 3};
    int *pz = z;
    pz = pz + 2;
    (*pz)++;
    printf("%d", *pz);
    return 0;
}
```

будет выведено значение _____

31. В результате работы программы:

```
int main() {
    int *a;
    a = 1050;
    printf("%d", a+2);
    return 0;
}
```

будет выведено значение _____.

32. В результате работы программы:

```
int main() {
    int a[] = {4, 2, 3, 1, 5};
    printf("%d", *(a+1) + *(a+3));
    return 0;
}
```

будет выведено значение _____.

33. В результате работы программы:

```
int main() {
    int a[] = {4, 2, 3, 1, 5};
    int *b = a + 2;
    printf("%d", b[-1]);
    return 0;
}
```

будет выведено значение _____.

34. Напишите функцию swap, которая может обменить два значения одинаковых произвольных типов. Пример вызова требуемой функции из main:

```
int main() {
    int a = 1, b = 2;
    char c = 3, d = 4;
    swap(&a, &b, sizeof(int));
    swap(&c, &d, sizeof(char));
    printf("%d, %d, %d, %d", a, b, c, d); // 2, 1, 4, 3
    return 0;
}
```

35. В языке С допустимо два обращения к полям структуры и объединений:

```
s.x

s->x
```

В каких случаях используется первый вариант обращения, в каких случаях – второй.

Опишите каким образом может быть заменен второй вариант обращения через первый.

36. В результате работы программы:

```
struct point1 {
    float x;
    char y;
    char z;
};

struct point2 {
    char x;
    float y;
    char z;
};


int main() {
    struct point1 p1;
    struct point2 p2;
    printf("%d", sizeof(p1));
    printf("%d", sizeof(p2));
}
```

будут выведены значения: _____.

37. В чем разница между структурами и объединениями?

38. В результате работы программы:

```
union t_u {
    int x;
    char y;
};

int main() {
    union t_u p;
    printf("%d", sizeof(p));
    return 0;
}
```

будет выведено: _____.

39. В результате работы программы:

```
union t_u {
    int x;
    char y;
};


int main() {
    union t_u p1;
    p1.x = 1 << 8;
    printf("%d", p1.y);
    return 0;
}
```

будет выведено: _____.

40. В результате работы программы:

```
struct point {
    unsigned char w:2;
    unsigned char x:2;
    unsigned char y:2;
    unsigned char z:2;
};


int main() {
    struct point p1;
    printf("%d", sizeof(p1));
    return 0;
}
```

будет выведено: _____.

41. В результате работы программы:

```
struct point1 {
    char x:2;
    char y:2;
};

struct point2 {
    unsigned char x:2;
    unsigned char y:2;
};


int main() {
    struct point1 p1;
    p1.x = 1;
    p1.y = 2;
    printf("%d, %d\n", p1.x, p1.y);

    struct point2 p2;
    p2.x = 1;
    p2.y = 2;
    printf("%d, %d", p2.x, p2.y);
    return 0;
}
```

42. Была запущена программа:

```
int main() {
    int a, b, c;
    a = b = c = 0;
    scanf("%d, %d", &a, &b);
    scanf("%d", &c);
    printf("%d %d %d", a, b, c);
    return 0;
}
```

и при приглашении на ввод в одной строке были введена строка “1 2 3”. В результате выполнения программы будут выведены следующие значения: _____.

43. В строке s записано произвольное вещественное число. Наличие части до точки и после точки гарантируется. Обменяйте на строке s местами части до точки и после точки. Например, в данном случае программа

```
int main() {
    char s[] = "1431.23";
    int a, b;
    // ваш код здесь
    //
    // использование других переменных кроме s, a и b
    // запрещено
    printf(s);
    return 0;
}
```

должна выдать: 23.1431.

44. В результате работы программы:

```
int main() {

    int x[][3] = {
            {1, 2},
            {4, 5, 6}
    };

    x[2][-1] = 10;
    x[-1][3] = 10;

    for (int i = 0; i < 2; i++) {
        for (int j = 0; j < 3; j++) {
            printf("%d ", x[i][j]);
        }
    }
    return 0;
}
```

будет выведено: _____.

45. Моя фантазия на вопросы почти закончилась, а место на листе – нет. Поэтому, последний вопрос: что будет выведено на экран в результате работы программы:

```
int main(int argc, char *argv[]) {
    register unsigned short int a[3] = {1, 2};
    printf("%d", 1[a]*2 + a[2]);
    return 0;
}
```

Ф.И.: _____

Группа: _____

Бланк ответов:

1	
2	
3	
4	
5	
6	
7	
8	
9	
10	
11	
12	
13	
14	
15	
16	
17	Вопрос с развернутым ответом
18	Вопрос с развернутым ответом
19	Вопрос с развернутым ответом
20	Вопрос с развернутым ответом
21	Вопрос с развернутым ответом
22	
23	
24	
25	Вопрос с развернутым ответом
26	
27	
28	Вопрос с развернутым ответом
29	
30	
31	
32	
33	
34	Вопрос с развернутым ответом
35	Вопрос с развернутым ответом
36	
37	Вопрос с развернутым ответом
38	
39	
40	
41	
42	
43	
44	
45	

Ключ:

1	3
2	4
3	255
4	128 -126 128
5	2
6	3
7	0 1 2 1
8	Ничего не будет выведено. Ошибка компиляции.
9	64
10	1
11	x = x & ~0xFF или x = (x >> 8) << 8
12	(a > b) ? a : b;
13	13
14	1, 0
15	4
16	5
17	Вопрос с открытым ответом.
18	
19	
20	
21	
22	10
23	11 12 20 20
24	Утечки нет. 40
25	Вопрос с открытым ответом.
26	10, 10
27	6
28	Вопрос с открытым ответом.
29	0, 1, 0
30	4
31	1058
32	3
33	4
34	Вопрос с открытым ответом.
35	Вопрос с открытым ответом.
36	8, 12
37	Вопрос с открытым ответом.
38	4
39	0
40	1
41	1,-2 1, 2
42	1 0 2
43	sscanf(s, "%d.%d", &a, &b); sprintf(s,"%d.%d", b, a);
44	10 2 0 4 5 10
45	4