

Вопрос 2

Что напечатает программа

```
int main()
{
    int value = 1;
    for (int i = 65; i < 256; ++i)
        value = (value * i) % 64;
    printf("%d", value);
    return 0;
}
```

Ваш ответ: **0**

Вопрос 3



Дано сбалансированное бинарное дерево поиска из $N \cdot 2^N$ вершин. Каково худшее время поиска заданного ключа ?

Ваш ответ:

- ☐ $O(N)$
- ☐ $O(N \cdot \log(N))$
- ☐ $O(\log(N))$
- ☐ $O(2^N)$
- ☐ $O(N + \log(N))$

Ответить

Пропустить

$N \cdot \log(N)$

Вопрос 4



Какие из приведенных ниже операций будут выполнены в очереди с приоритетом (heap) не за $\Theta(\log N)$?

$$f(n) = \Theta(g(n)) \Leftrightarrow \exists a, b > 0, N : \forall n > N : f(n) < a \cdot g(n), f(n) > b \cdot g(n)$$

Ваш ответ:

- ☐ Нахождение минимума (максимума) без удаления его из кучи
- ☐ Удаление элемента по его ключу
- ☐ Нахождение минимума (максимума) с последующим удалением его из кучи
- ☐ Добавление элемента
- ☐ Получение значения элемента по его ключу

Ответить

Пропустить

Все кроме удаления и добавления

Вопрос 6

Структура, представляющую собой закольцованный массив, может служить для буферизации ввода-вывода. Скажите, что напечатает приведенный ниже код на языке C, включающий в себя специфичную реализацию этой структуры?

```
#include <stdio.h>
#include <stdlib.h>
struct buffer
{
    int begin, end, size;
    char *buf;
};
typedef struct buffer buffer;
buffer* create_buffer(int size)
{
    buffer *b = calloc(1, sizeof(*b));
    b->begin = b->end = 0;
    b->size = size;
    b->buf = calloc(size, sizeof(b->buf[0]));
    return b;
}
int write_buffer(buffer *b, void *data, int size)
{
    char *d = (char *)data;
    int i;
    for (i = 0; i < size && b->begin != (b->end + 1) % b->size; ++i)
    {
        b->buf[b->end] = d[i];
        b->end = (b->end + 1) % b->size;
    }
    return i;
}
int read_buffer(buffer *b, void *data, int size)
{
    char *d = (char *)data;
    int i;
    for (i = 0; i < size && b->begin != b->end; ++i)
    {
        d[i] = b->buf[b->begin];
        b->begin = (b->begin + 1) % b->size;
    }
    return i;
}
int main(void)
{
```

```
buffer* b = create_buffer(10);
for (char i = 0; i <= 15; ++i)
    write_buffer(b, &i, sizeof(i));
short p;
while (read_buffer(b, &p, sizeof(p)) == sizeof(p))
{
    printf("%d ", (char)p);
}
return 0;
}
```

0 2 4 6

Вопрос 8



Два необычных шахматных коня договорились о встрече на шахматном поле $N \times M$. Для этого им нужно оказаться на одной и той же клетке в один момент времени. Необычность коней заключается в том, что ходят они не по очереди, а одновременно. Изначально один конь стоит в клетке (x_1, y_1) , а другой – в клетке (x_2, y_2) .

$$\begin{aligned} 0 &\leq x_1, x_2 \leq N - 1 \\ 0 &\leq y_1, y_2 \leq M - 1. \end{aligned}$$

Через сколько ходов произойдет долгожданная встреча? Решите задачу для:

$$\begin{aligned} N &= 10, M = 14 \\ x_1 &= 5, y_1 = 1 \\ x_2 &= 9, y_2 = 12. \end{aligned}$$

Если вы считаете, что встреча не произойдет, то впишите -1

Ваш ответ:

Ответить

Пропустить

Вопрос 11

Что будет выведено в результате выполнения следующей программы на языке C?

```
#include <stdio.h>
int foo(int n)
{
    for (int i = 2; i * i <= n; ++i)
        if (n % i == 0)
            return 1;
    return 0;
}
int main(void)
{
    int result = 0;
    for (int i = 100; i >= 50; --i)
        result += foo(i);
    printf("%d", result);
    return 0;
}
```

41

12

1

2

3

4

5

6

7

8

9

10

11

12

13

14

15

16

17

18

19

20

21

22

23

24

25

26

27

28

Вопрос 12

Какой из вариантов реализует стабильную сортировку по возрастанию?

```
void merge(int *a, int n, int m, int *buff)
{
    int i = 0;
    int j = 0;
    for ( ; i < n && j < m; )
    {
        /* code here*/
        {
            buff[i + j] = a[i];
            ++i;
        }
        else
        {
            buff[i + j] = a[j + n];
            ++j;
        }
    }
    for ( ; i < n; ++i)
        buff[i + j] = a[i];
    for ( ; j < m; ++j)
        buff[i + j] = a[j + n];
    memcpy(a, buff, sizeof(int) * (n + m));
}

void sort(int *a, int size, int *buff)
{
    if (size < 2)
        return;
    sort(a, size / 2, buff);
    sort(&a[size / 2], size - size / 2, &buff[size / 2]);
    merge(a, size / 2, size - size / 2, buff);
}
```

Ваш ответ:

☐ if (a[j + n] < a[i])

☐ if (a[j + n] <= a[i])

☒ if (a[j + n] > a[i])

☐ if (a[j + n] >= a[i])

Ответить

Пропустить

3

прос 13

Сколькими способами можно составить последовательность длины $n > 2$ из 0 и 1 так, чтобы никакие три нуля не стояли подряд?

Какая из приведенных ниже формул решает эту задачу?

```
#include <stdio.h>
int main()
{
    int n;
    scanf("%d", &n);
    int a[n + 1][4];
    a[2][0] = a[2][1] = a[2][2] = a[2][3] = a[2][4] = 1;
    for (int i = 3; i <= n; ++i)
    {
        /* code here */
    }
    printf("%d\n", a[n][0] + a[n][1] + a[n][2] + a[n][3]);
    return 0;
}
```

Ваш ответ:

```
a[i][0] = a[i][1] = a[i - 1][0] + a[i - 1][1];
a[i][2] = a[i - 1][1] + a[i - 1][3];
a[i][3] = a[i - 1][0];
```

```
a[i][0] = a[i][1] = a[i - 1][0] + a[i - 1][2];
a[i][2] = a[i - 1][1] + a[i - 1][3];
a[i][3] = a[i - 1][1];
```

```
a[i][0] = a[i][1] = a[i - 1][0] + a[i - 1][2];  
a[i][2] = a[i - 1][1];  
a[i][3] = a[i - 1][1] + a[i - 1][3];
```

```
a[i][0] = a[i][1] = a[i - 1][0] + a[i - 1][1];  
a[i][2] = a[i - 1][1] + a[i - 1][2];  
  
a[i][3] = a[i - 1][0];
```

Вопрос 17

На шахматной доске (8x8) стоит черная шашка. Сколькими способами она может попасть в дамки?

Черная шашка ходит по диагонали на одну клетку вниз-вправо или вниз-влево. Шашка проходит в дамки, если она попадает на нижнюю горизонталь.

Дополните код программы на языке C так, чтобы он решал эту задачу. На вход программе подается два числа от 1 до 8: номер столбца(считая слева) и номер строки(начиная снизу), где изначально стоит шашка.

```
#include <stdio.h>
#define MAX_AREA 8
int main()
{
    int x,y;
    int a[MAX_AREA + 2][MAX_AREA + 2];
    int sum = 0;
    for (int i = 0; i < MAX_AREA + 2; i++)
        for (int j = 0; j < MAX_AREA + 2; j++)
            a[i][j] = 0;
    scanf("%d %d", &x, &y);
    x = MAX_AREA - x;
    y = MAX_AREA - y;
    a[y][x] = 1;
    for (int i = y - 1; i >= 1; i--)
        for (int j = MAX_AREA; j >= 1; j--)
            /* code here */
    for (int j = 1; j <= MAX_AREA; j++)
        sum += a[1][j];
    printf("%d", sum);
    return 0;
}
```

Ваш ответ:

$a[i][j] = a[i - 1][j - 1] + a[i - 1][j + 1];$

$$a[i + 1][j + 1] = a[i + 2][j + 2] + a[i + 2][j];$$

$$a[i][j] = a[i + 1][j - 1] + a[i + 1][j - 1];$$

$$a[i][j] = a[i][j + 1] + a[i + 1][j];$$

$$a[i - 1][j] = a[i][j - 2] + a[i][j + 1];$$

Вопрос 21

Структура, представляющую собой закольцованный массив, может служить для буферизации ввода-вывода. Скажите, что напечатает приведенный ниже код на языке C, включающий в себя специфичную реализацию этой структуры?

```
#include <stdio.h>
#include <stdlib.h>
struct buffer
{
    int begin, end, size;
    char *buf;
};
typedef struct buffer buffer;
buffer* create_buffer(int size)
{
    buffer *b = calloc(1, sizeof(*b));
    b->begin = b->end = 0;
    b->size = size;
    b->buf = calloc(size, sizeof(b->buf[0]));
    return b;
}
int write_buffer(buffer *b, void *data, int size)
{
    char *d = (char *)data;
    int i;
    for (i = 0; i < size && b->begin != (b->end + 1) % b->size; ++i)
    {
        b->buf[b->end] = d[i];
        b->end = (b->end + 1) % b->size;
    }
    return i;
}
int read_buffer(buffer *b, void *data, int size)
{
    char *d = (char *)data;
    int i;
    for (i = 0; i < size && b->begin != b->end; ++i)
    {
        d[i] = b->buf[b->begin];
        b->begin = (b->begin + 1) % b->size;
    }
    return i;
}
int main(void)
{
```

```
buffer* b = create_buffer(40);
for (long long i = 0; i <= 10; ++i)
    write_buffer(b, &i, sizeof(i));
long long p;
while (read_buffer(b, &p, sizeof(p)) == sizeof(p))
{
    printf("%d ", (short)p);
    write_buffer(b, &p, sizeof(int));
}
return 0;
}
```

0 1 2 3 4 0 0 0

Вопрос 22

Что напечатает программа ?

```
struct pool_t
{
    int *data;
    int size;
    int tail;
};
void push(pool_t &pool, int value)
{
    pool.tail = (pool.tail + 1) % pool.size;
    pool.data[pool.tail] = value;
}
int main()
{
    pool_t pool = {new int[5], 5, 0};
    for (int i = 0; i < 10; ++i)
        push(pool, i);
    for (int i = 0; i < pool.size; ++i)
        printf("%d ", pool.data[i]);
    return 0;
}
```

Ваш ответ: 9 5 6 7 8

Что будет выведено в результате выполнения следующей программы на языке C?

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

#define MAX_N 15

int main(void)
{
    int a[MAX_N][MAX_N];
    for (int i = 0; i < MAX_N; ++i)
    {
        memset(a[i], 0, MAX_N * sizeof(a[i][0]));
        a[i][0] = a[i][i] = 1;
    }
    for (int i = 1; i < MAX_N; ++i)
        for (int j = 1; j < i; ++j)
            a[i][j] = a[i - 1][j] + a[i - 1][j - 1];
    printf("%d", a[8][5]);
    return 0;
}
```

56

опрос 27

Что напечатает программа в 7 строке

```
int main()
{
    int a = 1;
    int b = 2;
    for (int i = 0; i < 10; ++i)
    {
        int c = a + b;
        a = b;
        b = c;
        printf("%d\n", b);
    }
    return 0;
}
```

Ваш ответ: 55

Вопрос 30

Какой из приведенных ниже алгоритмов сортировки имеет лучшую оценку сложности в худшем случае для связного списка?

Ваш ответ: Сортировка слиянием

Вопрос 31

Что напечатает программа?

```
int foo(int a, int b)
{
    return b ? foo(b, a % b) : a;
}
int main()
{
    int a = 169;
    int b = 144;
    printf("%d\n", foo(a, b));
    return 0;
}
```

ОТВЕТ 1

Вопрос 35

Что будет напечатано в результате выполнения приведенной программы на языке C, если ей на вход были поданы числа 777 333?

```
#include <stdio.h>
long long func(long long a, long long n)
{
    if (n == 0)
        return 0;
    if (n & 1)
        return func(a, n & -2) + a;
    else
    {
        long long b = func(a, n >> 1);
        return b + b;
    }
}
int main(void)
{
    long long a, b;
    scanf("%lld %lld", &a, &b);
    printf("%lld\n", func(a, b));
    return 0;
}
```

}

258741

Вопрос 36

Что будет выведено в результате выполнения следующей программы на языке C?

```
#include <stdio.h>
int foo(int a, int b, int *x, int *y)
{
    if (a == 0)
    {
        *x = 0; *y = 1;
        return b;
    }
    int x1, y1;
    int d = foo(b % a, a, &x1, &y1);
    *x = y1 - (b / a) * x1;
    *y = x1;
    return d;
}
int main(void)
{
    int a = 11, b = 13;
    int x, y;
    int res = foo(a, b, &x, &y);
    printf("%d %d %d", x, y, res);
    return 0;
}
```

6 -5 1

Вопрос 39

Дополните код на языке C, в котором пропущена одна строка, так, чтобы он реализовывал очередь с приоритетом (heap).

```
#define MAX_SIZE 100
typedef struct
{
    int heap_size;
    int val[MAX_SIZE];
} heap;
int get_max(heap *h)
{
    return /* code here */;
}
void add_heap(heap *h, int x)
{
    int pos = ++h->heap_size;
    while (pos > 1 && x > h->val[pos >> 1])
    {
        h->val[pos] = h->val[pos >> 1];
        pos >>= 1;
    }
    h->val[pos] = x;
}
void extract_max(heap *h)
{
    int pos = 1;
    while ( 2 * pos < h->heap_size )
    {
        int idx_max = h->val[2 * pos] > h->val[2 * pos + 1]
            ? 2 * pos
            : 2 * pos + 1;
        if (h->val[h->val[h->heap_size]] >= h->val[idx_max])
            break;
        h->val[pos] = h->val[idx_max];
        pos = idx_max;
    }
    h->val[pos] = h->val[h->heap_size-];
}
```

Ваш ответ:


```
h->val[0];
```

```
h->val[1];
```

```
h->val[2];
```

```
h->val[h->heap_size];
```

```
h->val[h->heap_size - 1];
```

```
h->val[h->heap_size + 1];
```

Вопрос 40

Какие из предложенных вариантов приводят оценке времени порядка $O(N)$ для вставки N элементов в массив?

```
struct array_t
{
    int *data;
    int capacity;
    int size;
};
void add(array_t array, int value)
{
    if (size == capacity)
    {
        array.capacity = /* code here */;
        int *tmp = new int[array.capacity];
        memcpy(tmp, array.data, sizeof(int) * array.size);
        swap(tmp, array.data);
        delete [] tmp;
    }
    array.data[array.size] = value;
    ++array.size;
}
```

Ваш ответ:

array.capacity + 1

array.capacity + 2

array.capacity + array.size

`array.capacity * 3 / 2`

`array.capacity << 1`

Вопрос 41

Дополните код, чтобы получился алгоритм быстрой сортировки:

```
int partition(int *a, int size)
{
    int i = 0;
    /* code here */
    {
        if (a[size - 1] > a[j])
        {
            swap(a[j], a[i]);
            ++i;
        }
    }
    swap(a[i], a[size - 1]);
    return i;
}

void sort(int *a, int size)
{
    if (size < 2)
        return;
    int k = partition(a, size);
    sort(a, k);
    sort(&a[k + 1], size - k - 1);
}
```

Ваш ответ:

for (int j = 1; j < size; ++j)

for (int j = 2; j < size; ++j)

for (int j = 0; j + 1 < size; ++j)

```
for (int j = 1; j + 1 < size; ++j)
```

Вопрос 42

Как следует дополнить программу, чтобы она напечатала последовательность кубов натуральных чисел

```
int main()
{
    int a = 0;
    for (int i = 0; i < 10; ++i)
    {
        /* code here */
        printf("%d ", a);
    }
    return 0;
}
```

Ваш ответ:

`a = a + 3 * i * i;`

`a = a + 3 * i * i + 3 * i + 1;`

`a = a + 3 * i * i - 3 * i + 1;`

`a = a + i * i;`

```
a = a + i * i - i + 1;
```

```
a = a + i * i + i - 1;
```

Вопрос 28



Пусть есть структура данных, поддерживающая операции:

CREATE(N) - создать структуру, которая будет отвечать на запросы на полуинтервале [0,N). Изначально все элементы из [0,N) равны 0.

SET(L, R, val) - присвоить значение val всем элементам из [L, R)

GET_SUM(L, R) - возвращает сумму элементов с целыми индексами из [L, R)

ADD(L, R, val) - прибавить ко всем элементам из [L, R) значение val

Что будет выведено в результате выполнения следующего псевдокода? (функция PRINT выводит число и пробел после него)

```
CREATE(15)
  FOR I = 0 TO 13
    FOR J = 1 TO 14
      ADD(I - 1, I + 1, I * (((I % 2) * 2) - 1))
    SET(I, I + 2, GET_SUM(I, I + 2))
  PRINT(GET_SUM(0, 15))
PRINT(GET_SUM(1, 3))
PRINT(GET_SUM(4, 8))
PRINT(GET_SUM(6, 12))
```

Ваш ответ:

Ответить

Пропустить