

【华为OD机考 统一考试机试C卷】GPU 调度/执行时长 (C++ Java JavaScript Python C语言)

华为OD机考:统一考试 C卷 + D卷 + B卷 +A卷

目前在考C卷，经过两个月的收集整理，**C卷真题已基本整理完毕**

抽到原题的概率为2/3到3/3，**也就是最少抽到两道原题。请注意：大家刷完C卷真题，最好要把B卷的真题刷一下，因为C卷的部分真题来自B卷。**

另外订阅专栏还可以联系笔者开通在线 **OJ** 进行刷题，提高刷题效率。

真题目录：华为OD机考机试 真题目录 (C卷 + D卷 + B卷 + A卷) + 考点说明

专栏：2023华为OD机试(B卷+C卷+D卷) (C++JavaJSPy)

华为OD面试真题精选：华为OD面试真题精选

在线OJ：点击立即刷题，模拟真实机考环境

题目描述

为了充分发挥GPU[算力]，需要尽可能多的将任务交给GPU执行，现在有一个任务数组，数组元素表示在这1秒内新增的任务个数且每秒都有新增任务。

假设GPU最多一次执行n个任务，一次执行耗时1秒，在保证GPU不空闲情况下，最少需要多长时间执行完成。

输入描述

- 第一个参数为GPU一次最多执行的任务个数，取值范围[1, 10000]
- 第二个参数为任务数组长度，取值范围[1, 10000]
- 第三个参数为任务数组，数字范围[1, 10000]

输出描述

执行完所有任务最少需要多少秒。

用例

输入	3 5 1 2 3 4 5
输出	6
说明	一次最多执行3个任务，最少耗时6s

输入	4 5 5 4 1 1 1
输出	5
说明	一次最多执行4个任务，最少耗时5s

C++

```

1  #include <iostream>
2  using namespace std;
3
4  int main() {
5      int maxTasks;
6      cin >> maxTasks;
7
8      int taskArrLen;
9      cin >> taskArrLen;
10
11     int* taskArr = new int[taskArrLen];
12     for (int i = 0; i < taskArrLen; i++) {
13         cin >> taskArr[i];
14     }
15
16     int currentTasks = 0;
17     int time = 0;
18     int index = 0;
19
20     while (currentTasks != 0 || index != taskArrLen) {
21         if (index < taskArrLen) {
22

```

```

23         currentTasks += taskArr[index];
24         index++;
25     }
26
27     currentTasks -= maxTasks;
28
29     if (currentTasks < 0) currentTasks = 0;
30
31     time++;
32 }
33
34 cout << time << endl;
35
36 delete[] taskArr;
37
38 return 0;
39 }

```

javascript

```

1  const readline = require('readline');
2
3  const rl = readline.createInterface({
4      input: process.stdin,
5      output: process.stdout
6  });
7
8  let maxTasks = 0;
9  let taskArrLen = 0;
10 let taskArr = [];
11 let currentTasks = 0;
12 let time = 0;
13 let index = 0;
14
15 rl.on('line', (line) => {
16     if (maxTasks === 0) {
17         maxTasks = parseInt(line.trim());
18     } else if (taskArrLen === 0) {
19         taskArrLen = parseInt(line.trim());
20     }
21 });

```

```

20     } else {
21
22         taskArr = line.split(" ").slice(0, taskArrLen).map((ele) => parseInt(ele));
23
24     }
25 });
26
27 rl.on('close', () => {
28     while (currentTasks !== 0 || index !== taskArr.length) {
29         if (index < taskArr.length) {
30             currentTasks += taskArr[index];
31             index++;
32         }
33         currentTasks -= maxTasks;
34         if (currentTasks < 0) currentTasks = 0;
35         time++;
36     }
37
38     console.log(time);
39 });
40

```

java

```

1
2 import java.util.Scanner;
3
4 public class Main {
5     public static void main(String[] args) {
6         Scanner in = new Scanner(System.in);
7
8         // 输入GPU一次最多执行的任务个数
9         int maxTasks = in.nextInt();
10
11         // 输入任务数组长度
12         int taskArrLen = in.nextInt();
13
14         // 输入任务数组
15         int[] taskArr = new int[taskArrLen];
16         for (int i = 0; i < taskArrLen; i++) {
17

```

```

17         taskArr[i] = in.nextInt();
18     }
19
20     // 当前任务个数
21     int currentTasks = 0;
22
23     // 执行任务所需时间
24     int time = 0;
25
26     // 任务数组索引
27     int index = 0;
28
29     // 当前任务个数不为0或任务数组未遍历完时继续执行
30     while (currentTasks != 0 || index != taskArr.length) {
31         // 如果任务数组还有任务未遍历
32         if (index < taskArr.length) {
33             // 将新增的任务加到当前任务个数中
34             currentTasks += taskArr[index];
35             index++;
36         }
37
38         // 执行一次最多的任务个数
39         currentTasks -= maxTasks;
40
41         // 如果当前任务个数小于0, 将其置为0
42         if (currentTasks < 0) currentTasks = 0;
43
44         // 执行任务所需时间加1
45         time++;
46     }
47
48     // 输出执行完所有任务所需的最少时间
49     System.out.println(time);
50 }
51 }

```

python

```

1 maxTasks = int(input())
2 taskArrLen = int(input())
3

```

```

3 taskArr = list(map(int, input().split()))
4
5
6 currentTasks = 0
7 time = 0
8 index = 0
9
10 while currentTasks != 0 or index != len(taskArr):
11     if index < len(taskArr):
12         currentTasks += taskArr[index]
13         index += 1
14
15     currentTasks -= maxTasks
16     if currentTasks < 0:
17         currentTasks = 0
18
19     time += 1
20
21 print(time)

```

C语言

```

1 #include <stdio.h>
2 #include <stdlib.h>
3
4 int main() {
5     int maxTasks;
6     scanf("%d", &maxTasks);
7
8     int taskArrLen;
9     scanf("%d", &taskArrLen);
10
11     // 使用malloc分配动态数组
12     int* taskArr = (int*)malloc(taskArrLen * sizeof(int));
13     for (int i = 0; i < taskArrLen; i++) {
14         scanf("%d", &taskArr[i]);
15     }
16
17     int currentTasks = 0;
18     int time = 0;
19
20

```

```

19     int index = 0;
20
21     // 当前任务不为0或者还有任务未添加到currentTasks时, 继续执行
22     while (currentTasks != 0 || index != taskArrLen) {
23         if (index < taskArrLen) {
24             // 将新增的任务添加到currentTasks
25             currentTasks += taskArr[index];
26             index++;
27         }
28
29         // 每秒执行maxTasks个任务
30         currentTasks -= maxTasks;
31
32         // 如果执行后任务数为负, 则置为0
33         if (currentTasks < 0) currentTasks = 0;
34
35         // 时间增加
36         time++;
37     }
38
39     printf("%d\n", time);
40
41     // 释放动态分配的内存
42     free(taskArr);
43
44     return 0;
45 }

```

完整用例

用例1

```

1 | 3
2 | 5
3 | 1 2 3 4 5

```

用例2

1	4
2	5
3	5 4 1 1 1

用例3

1	2
2	3
3	1 1 1

用例4

1	1
2	5
3	1 1 1 1 1

用例5

1	5
2	4
3	1 2 3 4

用例6

1	10
2	10
3	1 2 3 4 5 6 7 8 9 10

用例7

1	7
2	6
3	3 4 5 6 7 8

用例8

1	100
2	10
3	

用例9

1	5
2	8
3	1 1 1 1 1 1 1 1

用例10

1	3
2	10
3	2 2 2 2 2 2 2 2 2 2

文章目录

华为OD机考:统一考试 C卷 + D卷 + B卷 +A卷	
题目描述	
输入描述	
输出描述	
用例	
C++	
javascript	
java	
python	
C语言	
完整用例	
用例1	
用例2	
用例3	
用例4	
用例5	
用例6	
用例7	
用例8	

用例9
用例10

机考真题

华为OD



CSDN @算法大师