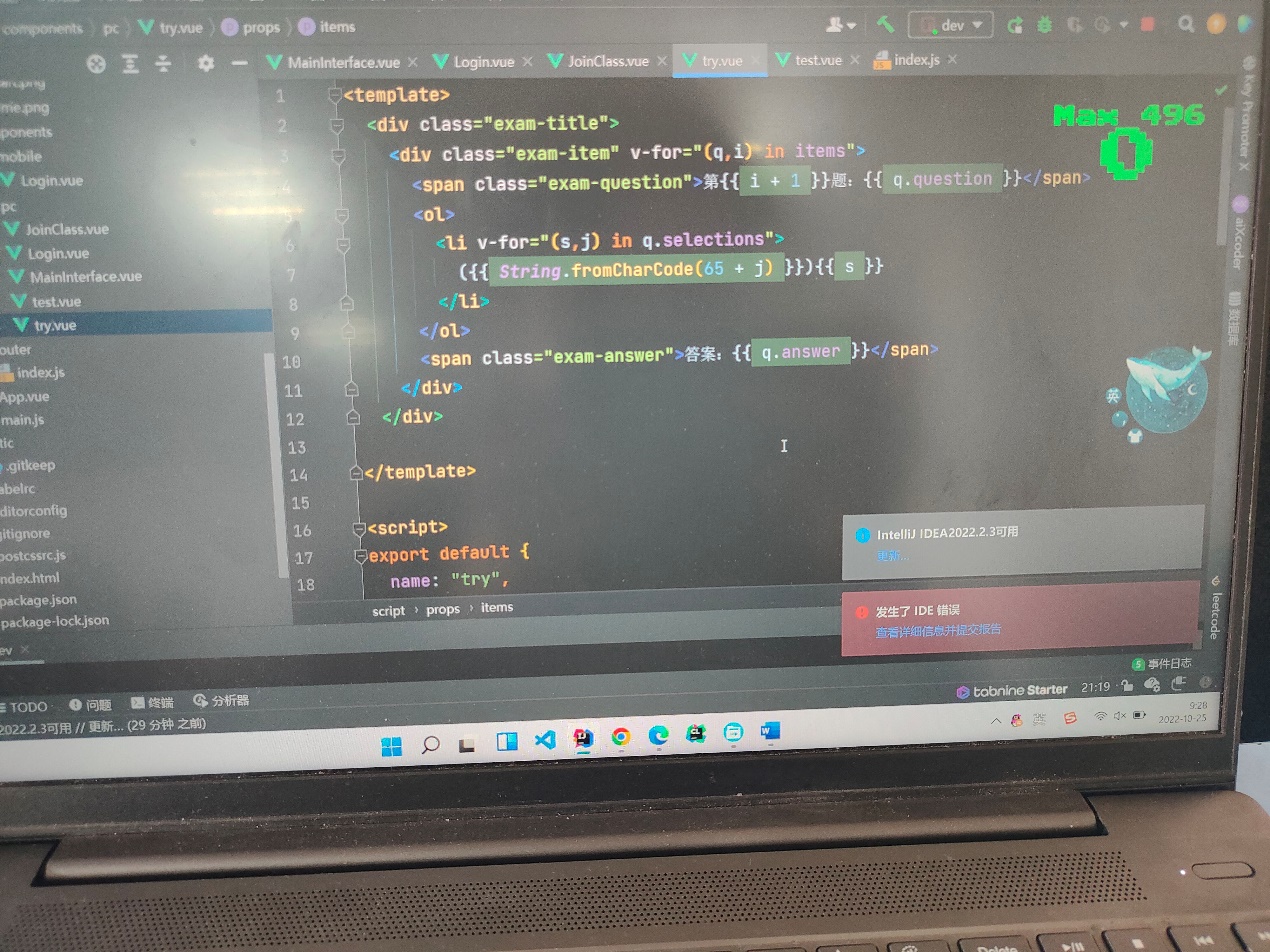
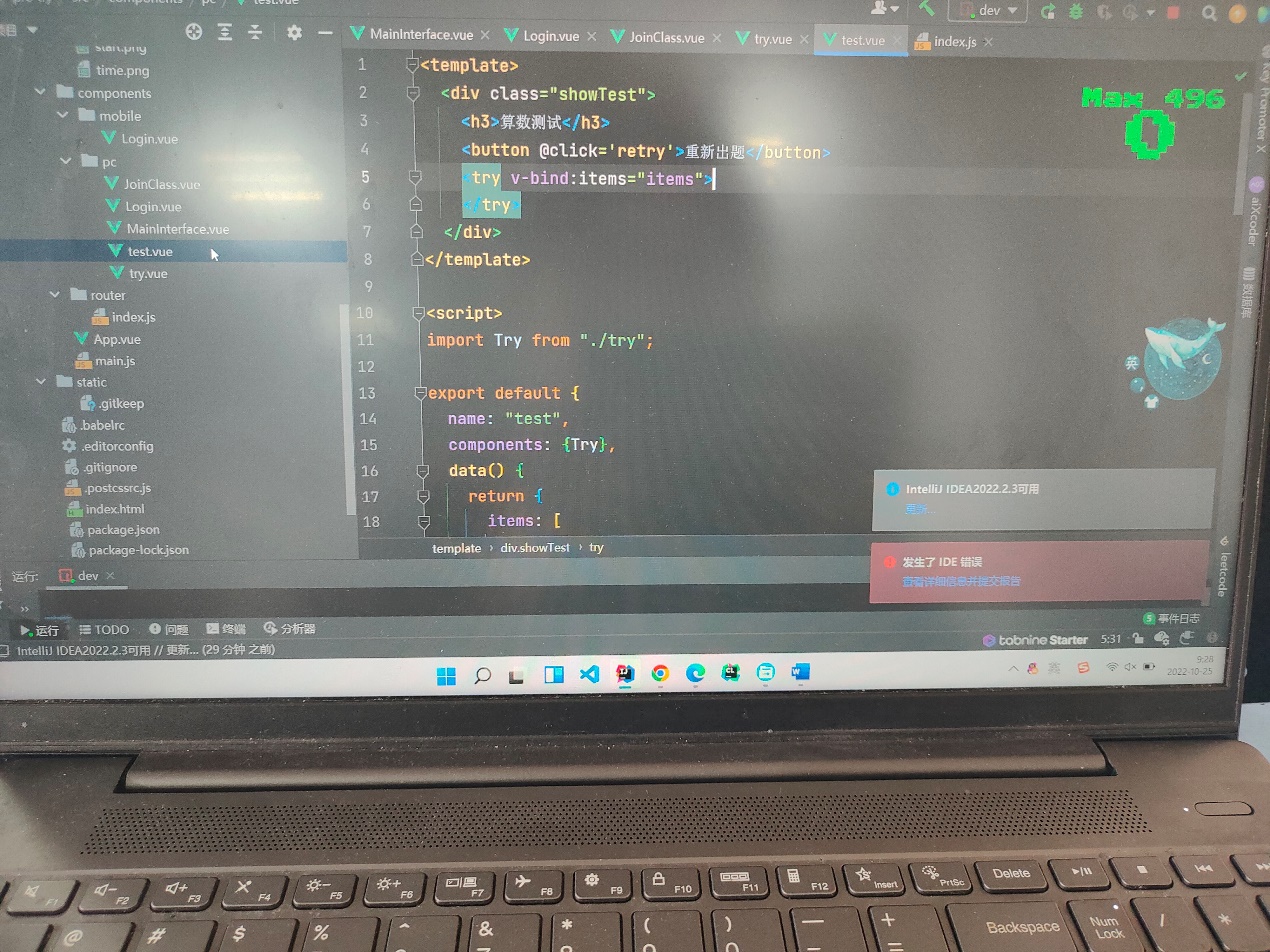
20201309-洪毅凡

9点28第一次拍照

子组件try.vue

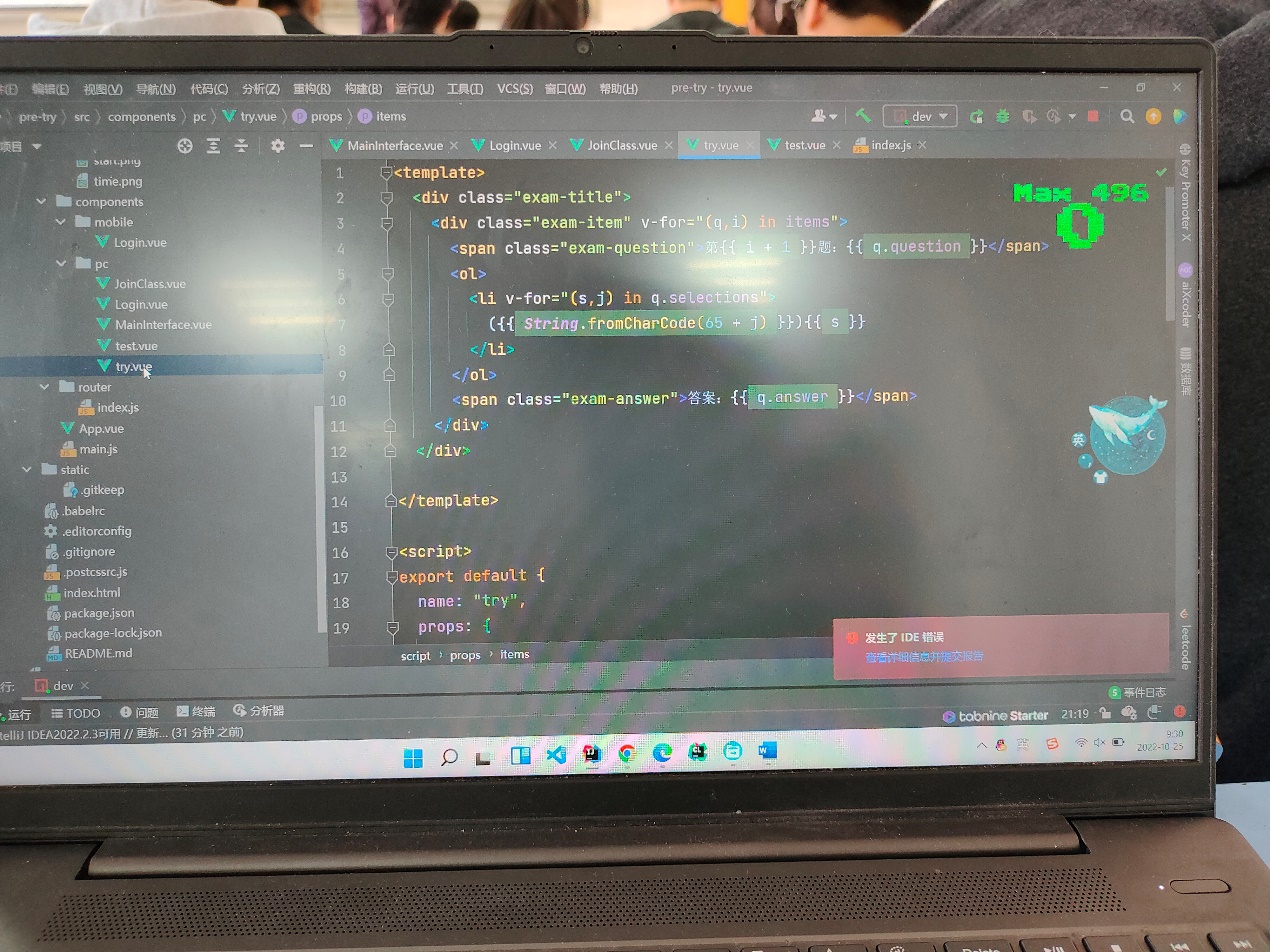


父组件test.vue

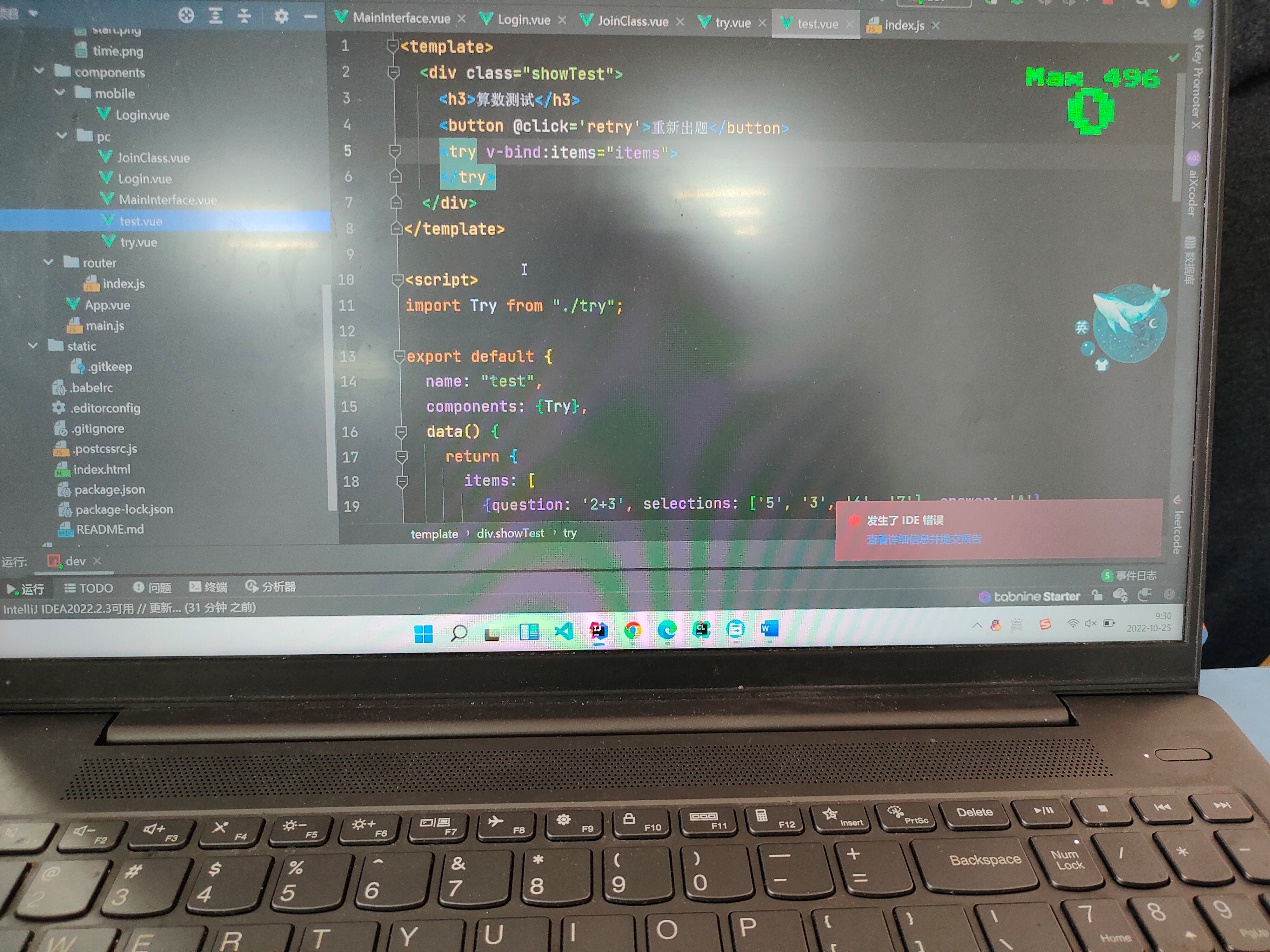


9点30第二次拍照

子组件try.vue

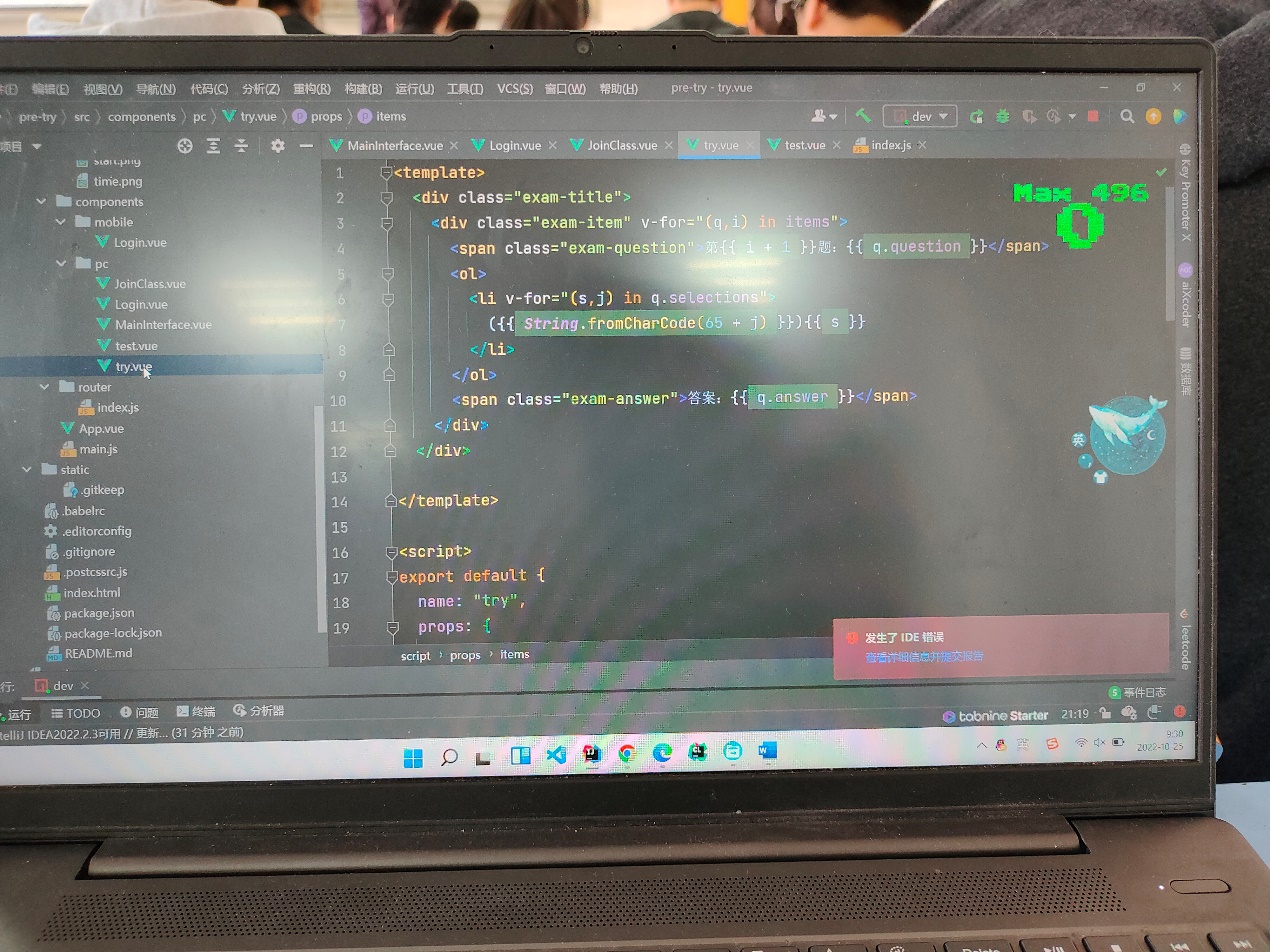


父组件test.vue

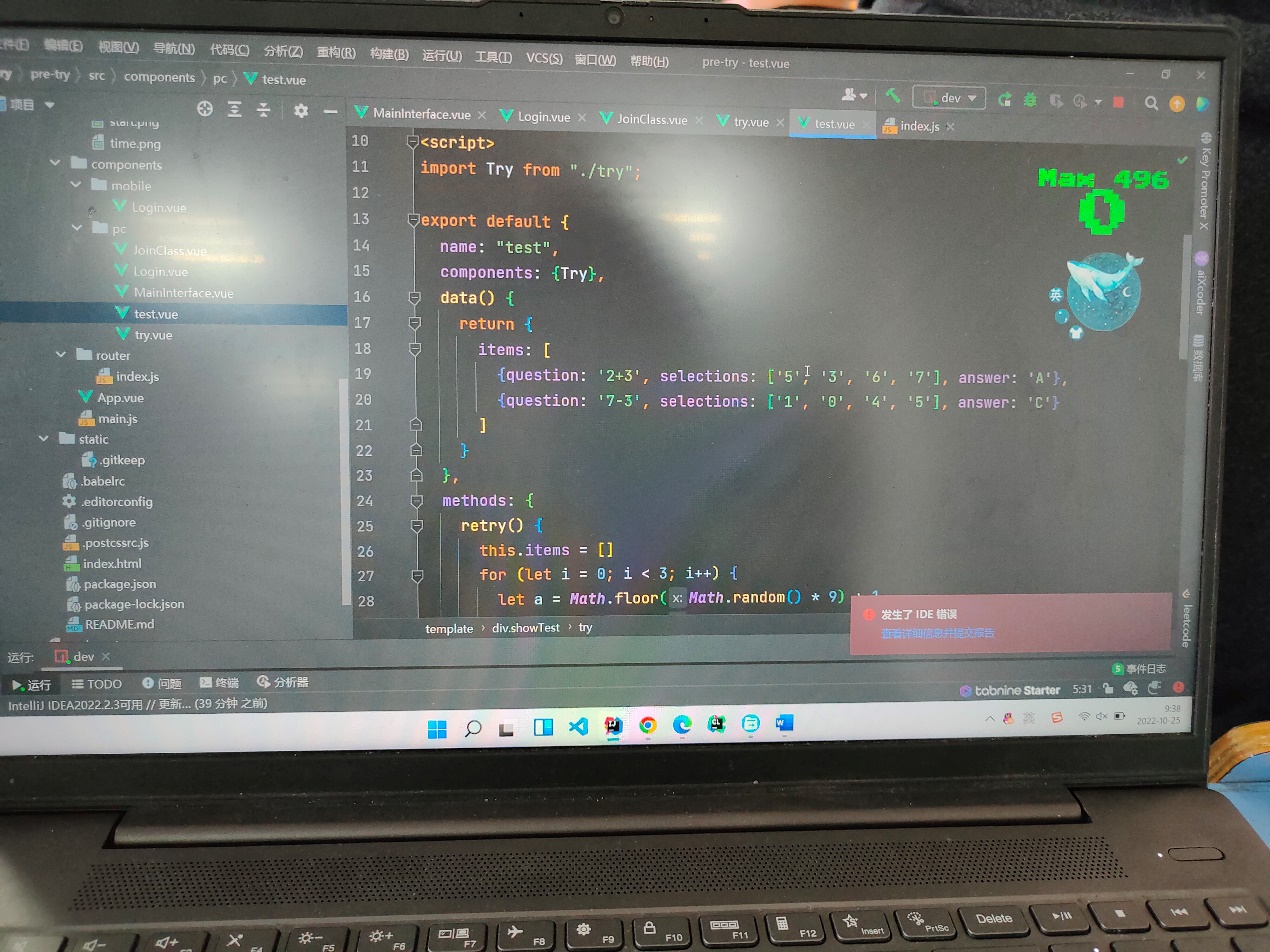


9点38第三次拍照

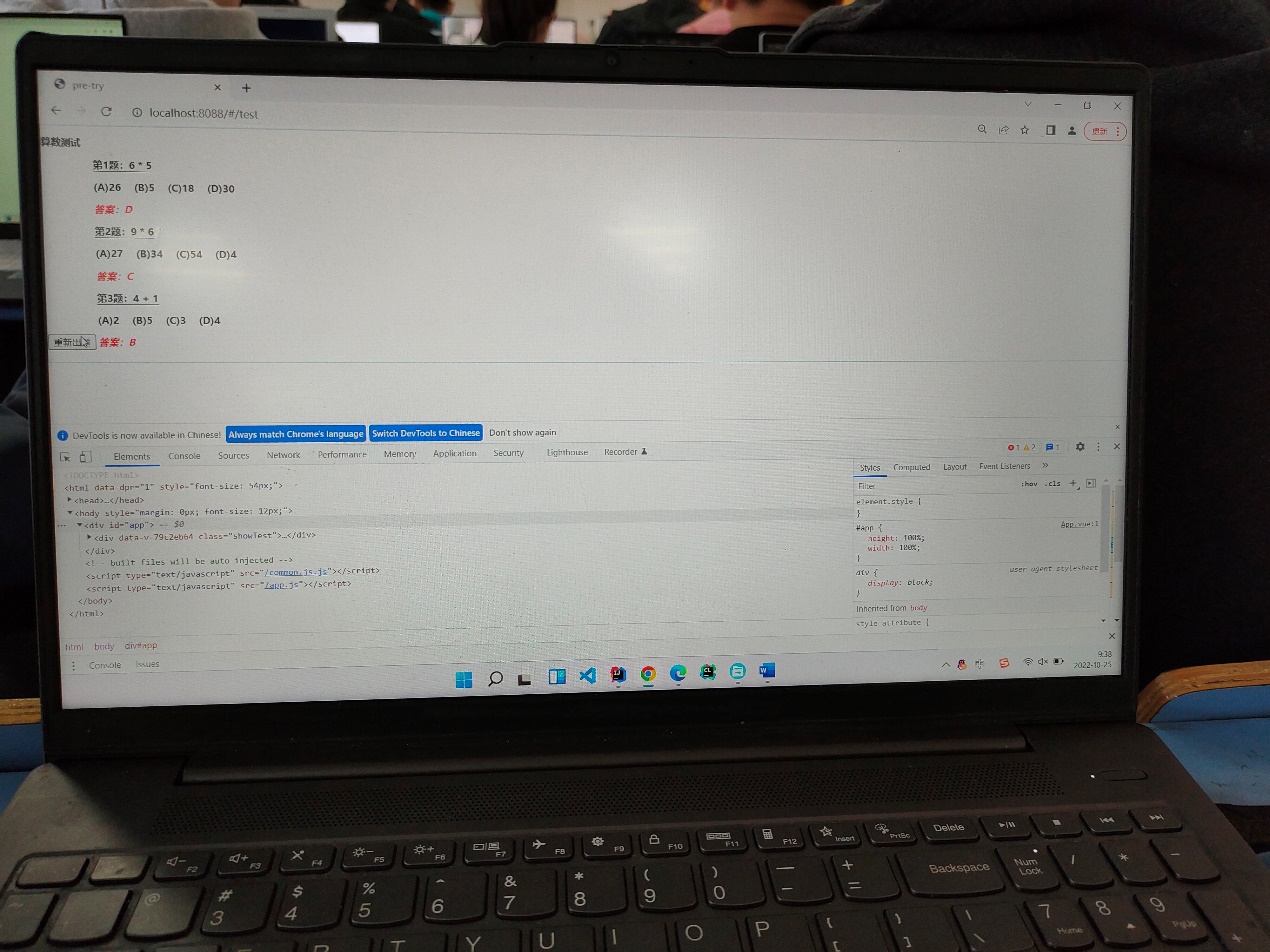
子组件try.vue

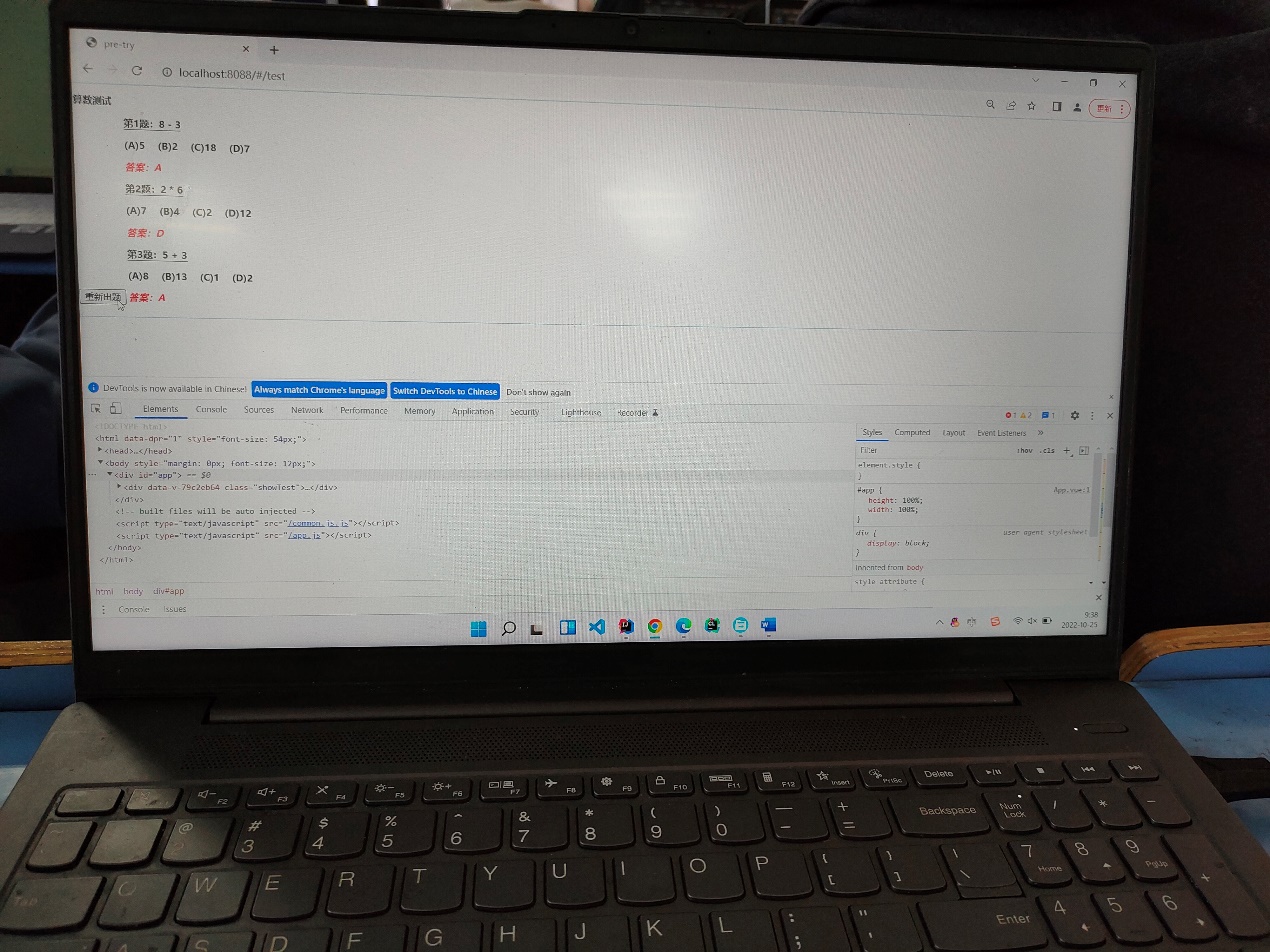


父组件test.vue



运行界面展示





子组件 try.vue

<template>  
 <div class="exam-title">  
 <div class="exam-item" v-for="(q,i) in items">  
 <span class="exam-question">第{{ i + 1 }}题：{{ q.question }}</span>  
 <ol>  
 <li v-for="(s,j) in q.selections">  
 ({{ String.fromCharCode(65 + j) }}){{ s }}  
 </li>  
 </ol>  
 <span class="exam-answer">答案：{{ q.answer }}</span>  
 </div>  
 </div>  
  
</template>  
  
<script>  
export default {  
 name: "try",  
 props: {  
 items: { //这个就是父组件中子标签自定义名字  
 type: ***Array***,  
 required: true  
 }  
 }  
}  
</script>  
  
<style scoped>  
.exam-title {  
 display: inline-block;  
 font-size: larger;  
 font-weight: bolder;  
 margin: 0 50px 10px 0;  
}  
  
.exam-item {  
 margin-bottom: 20px;  
}  
  
ol {  
 padding: 0;  
}  
  
ol li {  
 display: inline;  
 margin-right: 20px;  
}  
  
.exam-question {  
 border-bottom: 1px solid black;  
}  
  
.exam-answer {  
 color: red;  
 font-style: italic;  
}  
</style>

父组件test.vue

<template>  
 <div class="showTest">  
 <h3>算数测试</h3>  
 <button @click='retry'>重新出题</button>  
 <try v-bind:items="items">  
 </try>  
 </div>  
</template>  
  
<script>  
import Try from "./try";  
  
export default {  
 name: "test",  
 components: {Try},  
 data() {  
 return {  
 items: [  
 {question: '2+3', selections: ['5', '3', '6', '7'], answer: 'A'},  
 {question: '7-3', selections: ['1', '0', '4', '5'], answer: 'C'}  
 ]  
 }  
 },  
 methods: {  
 retry() {  
 this.items = []  
 for (let i = 0; i < 3; i++) {  
 let a = ***Math***.floor(***Math***.random() \* 9) + 1  
 let b = ***Math***.floor(***Math***.random() \* 9) + 1  
 let ops = [' + ', ' - ', ' \* ']  
 let op = ops[***Math***.floor(***Math***.random() \* 3)]  
 let question = a + op + b  
 let r = eval(question)  
 let selections = []  
 let answer = 'A'  
 for (let j = 0; j < 4; j++) {  
 let n = ***Math***.floor(***Math***.random() \* a \* b + 1)  
 while (selections.indexOf(n) >= 0)  
 n++  
 selections.push(n)  
 }  
 let k = selections.indexOf(r)  
 if (k < 0) {  
 k = ***Math***.floor(***Math***.random() \* 4)  
 selections[k] = r  
 }  
 answer = ***String***.fromCharCode(65 + k)  
 this.items.push({  
 question: question, selections: selections, answer: answer  
 })  
 }  
 }  
 }  
}  
</script>  
  
<style scoped>  
.showTest {  
 border: 1px skyblue solid;  
}  
</style>

还需要在路由index.js中挂载父组件test.vue，我是直接用两个组件完成，一个父组件用于承载子组件，子组件用来展示题目，题目的随机产生发生在父组件，此时需要进行组件通信，把产生的随机结果发送到子组件，也就是很通常的单页面由多个组件组成，需要进行组件间的通信。所以第一步是实现让子组件能展示出题目，题目答案的，然后向外暴露，接着在父组件中引入子组件，将父组件挂载到全局路由，在父组件中写出能产生随机题目的代码，并把题目封装在一个数组当中，通过v-bind传递这个数组到子组件，子组件写好prop属性去接收，修改子组件产生题目代码，通过v-for去循环整个数组。