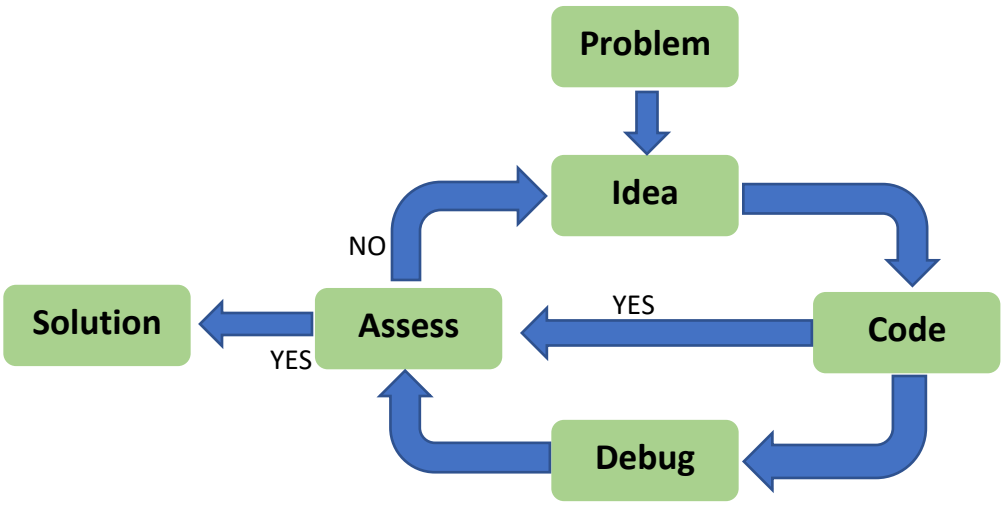
# مباحث ویژه در برنامه نویسی

نام و نام خانوادگی : حامد عسکری

تمرین : با روش رویکرد برنامه نویسی یک مسئله را حل کنید.

فلوچارت




1 : ابتدا از مرحله اول **problem** شروع میکنیم : **چگونه** یک کد بنویسیم که از یک آرایه اعداد زوج را برای ما پیدا کند و روشی سریع باشد؟ (با زبان JS) .در این مرحله ابتدا باید زبان آن مسئله را بدانیم سپس در آن تخصص داشته باشیم و درک کلی از آن هم داشته باشیم .

2 : بعد از درک کامل مسئله به روش های مختلف حل آن فکرمیکنیم و سعی میکنیم بهینه ترین را با توجه به نیاز مسئله در این لحظه انتخاب کنیم . در این مرحله درک کاملی از روش حل مسئله داریم.

میدانیم که هر عدد باقی مانده اش تقسیم بر دو اگر صفر شود آن عدد زوج میباشد. ایده ها و روش هایی که به دست آمد در این لحظه سه روش متفاوت هستند که ما یکی از آن هارا با توجه به نیاز برنامه یعنی سریع بودن انتخاب میکنیم.

3 : **Code** در این مرحله ما میخواهیم یکی از راه حل های انتخاب شده را تست کنیم در مرحله قبل سه روش به دست آمد.

کد روش اول انتخاب شده را مینوسیم : با استفاده از متد و عمل **تقسیم**.

```
const numbers = [1, 2, 3, 4, 5, 6, 7, 8, 9, 10];

const evenNumbers = numbers.filter(num => num % 2 === 0);

console.log(evenNumbersModulus); // [2, 4, 6, 8, 10]
```

در این روش از متد filter در جاوا اسکریپت استفاده شده است که تمام آرایه را پیمایش میکند و هر عنصر را تقسیم به دو میکند اگر باقی مانده صفر شود آن را در یک آرایه جدید ذخیره میکند و همین را ادامه میدهد تا یک آرایه از اعداد زوج به ما برگرداند.

4 : **Debug** در این مرحله اگر در اجرای کد مشکلی وجود نداشت به مرحله بعدی میرویم.

5 : **Assess** در این مرحله باید ارزیابی کنیم ما سه روش پیداکردیم پس باید روش های دیگر هم تست شوند در اینجا سرعت runtime برنامه را یادداشت کرده و سراغ راه حل بعدی میرویم تا در این مرحله بتوانیم به سریع ترین آن دست پیداکنیم پس دوباره به قسمت 2 idea برمیگردیم و روش بعدی را بررسی میکنیم

و پس از مشخص شدن روش درست پیاده سازی آن باز سراغ 3 code میرویم و کد روش جدید را مینویسیم درصورت نبود باگ به مرحله ارزیابی میرویم و این کد جدید را با نتایج runtime قبلی مقایسه میکنیم.

**همین مراحل** را تا رسیدن به سریع ترین جواب ادامه میدهیم.

روش دوم ما : انجام مقایسه به صورت بیتی فقط آخرین بیت را مقایسه میکند.

```
const evenNumbersBitwise = numbers.filter(num => (num & 1) === 0);
```

روش سوم ما : استفاده از حلقه for

```
const evenNumbersLoop = [];

for (let i = 0; i < numbers.length; i++) {
    if (numbers[i] % 2 === 0) {
        evenNumbersLoop.push(numbers[i]);
    }
}
```

| روش | سرعت | بهینه بودن | سادگی کد |
|---|---|---|---|
| عملگر % | خوب | مناسب | ساده |
| عملگر AND | سریع | عالی | کمتر شناخته |
| حلقه for | متوسط | خوب | کمی پیچیده |

6 : **Solution** در این مرحله با بررسی نتایج با توجه به نیاز های مسئله بهترین روش را انتخاب میکنیم که در این مورد میشود **روش عملگر AND** چون برای ما سرعت مهم بود.

منابع : perplexity.ai