

ALGORITMOS

ORDENAÇÃO

MERGE

GULOSOS

LÁ ELE

Douglas Toledo

NOVEMBRO 2023

Entendendo Ordenação e Algoritmos Gulosos >>>

A ordenação na computação é uma organização de elementos em uma sequência específica com base em critérios como ordem numérica, alfabética, cronológica, entre outros. Sua importância é fundamental em algoritmos, sistemas e processamento de dados, proporcionando eficiência, melhor desempenho em operações, facilitando a busca, manipulação e apresentação de informações coerentes, além de ser essencial em diversas áreas, desde bancos de dados até jogos e aplicativos da web. A escolha do algoritmo de ordenação correto pode impactar significativamente o desempenho de um sistema, tornando a ordenação uma ferramenta essencial na computação

Existem vários algoritmos de ordenação, cada um com suas próprias características, eficiência e complexidade:

- 01** Quick Sort - Algoritmo de Ordenação rápida.
- 02** Merge Sort - Algoritmo de Ordenação por mistura.
- 03** Selection Sort - Algoritmo de Ordenação por seleção.
- 04** Insertion Sort - Algoritmo de Ordenação por Inserção.
- 05** Bolha (bubble sort)

Cada um desses algoritmos possui vantagens e desvantagens em relação à velocidade de execução, utilização de memória e desempenho em diferentes cenários.

Entendendo Ordenação e Algoritmos Gulosos ➤➤

Os algoritmos gulosos, também conhecidos como algoritmos vorazes, são estratégias de resolução de problemas em que, em cada etapa, faz-se uma escolha que parece ser a melhor naquele momento, sem considerar possíveis consequências futuras. Ou seja, em cada passo, o algoritmo faz a escolha localmente ótima esperando que ela leve à solução global ótima.

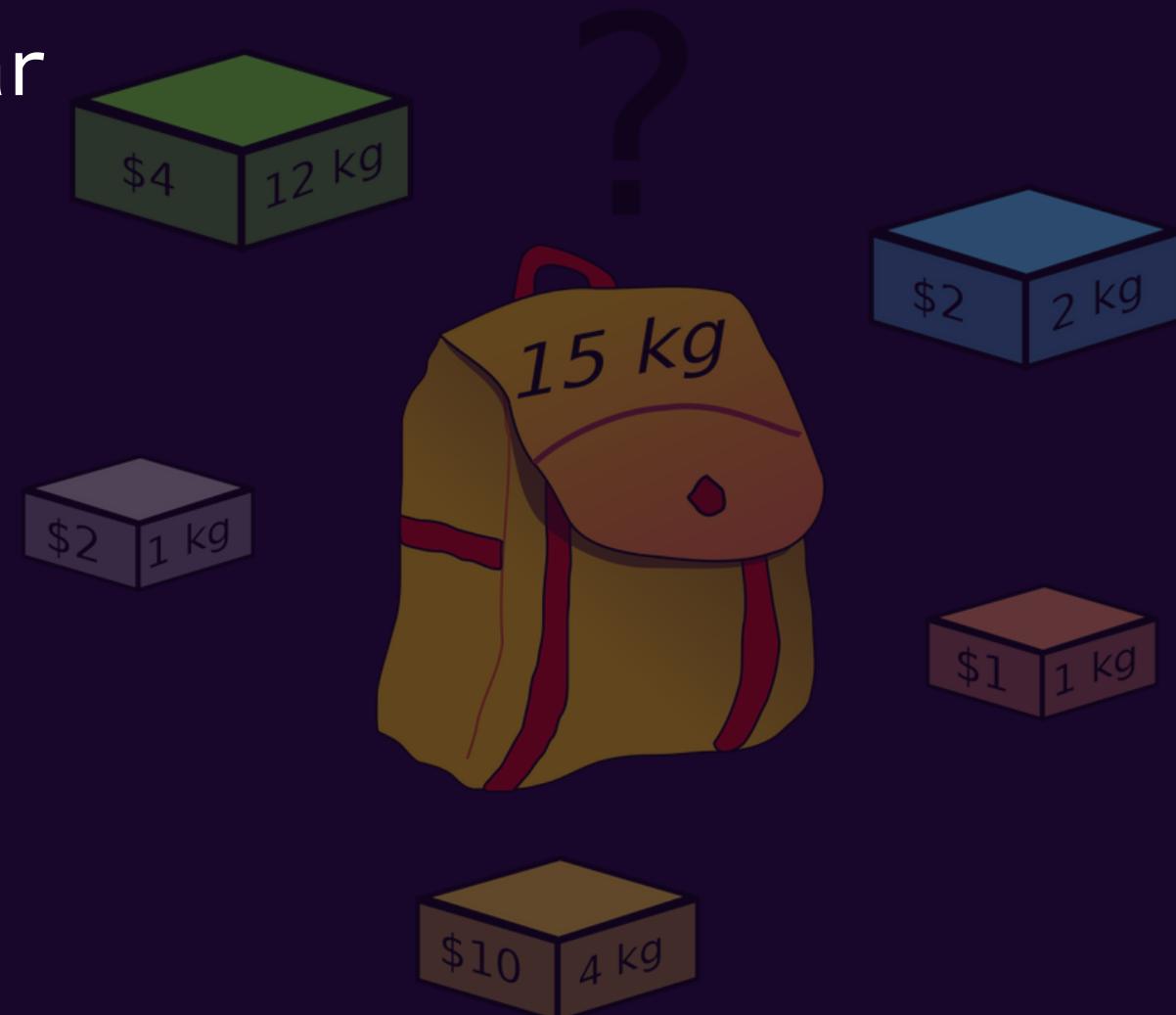
A principal característica dos algoritmos gulosos é a tomada de decisão localmente ótima em cada estágio do problema, sem revisar ou desfazer as escolhas feitas anteriormente. Eles geralmente são fáceis de implementar e são eficientes em termos de tempo de execução.

- 01 Algoritmo de Dijkstra
- 02 Algoritmo de Kruskal
- 03 Algoritmo de Huffman
- 04 Algoritmo de Seleção de Atividades

PROBLEMA DA MOCHILA BINÁRIA

Diferente da **MOCHILA FRACIONADA**, onde você pode colocar um pedaço do objeto dentro da mochila...
(ex.: ouro em pó)

Na **MOCHILA BINÁRIA** os objetos não podem ser particionados
(ou estarão dentro da mochila ou fora, 0 OU 1)
(ex.: ouro em barra)



PROBLEMA DA MOCHILA BINÁRIA

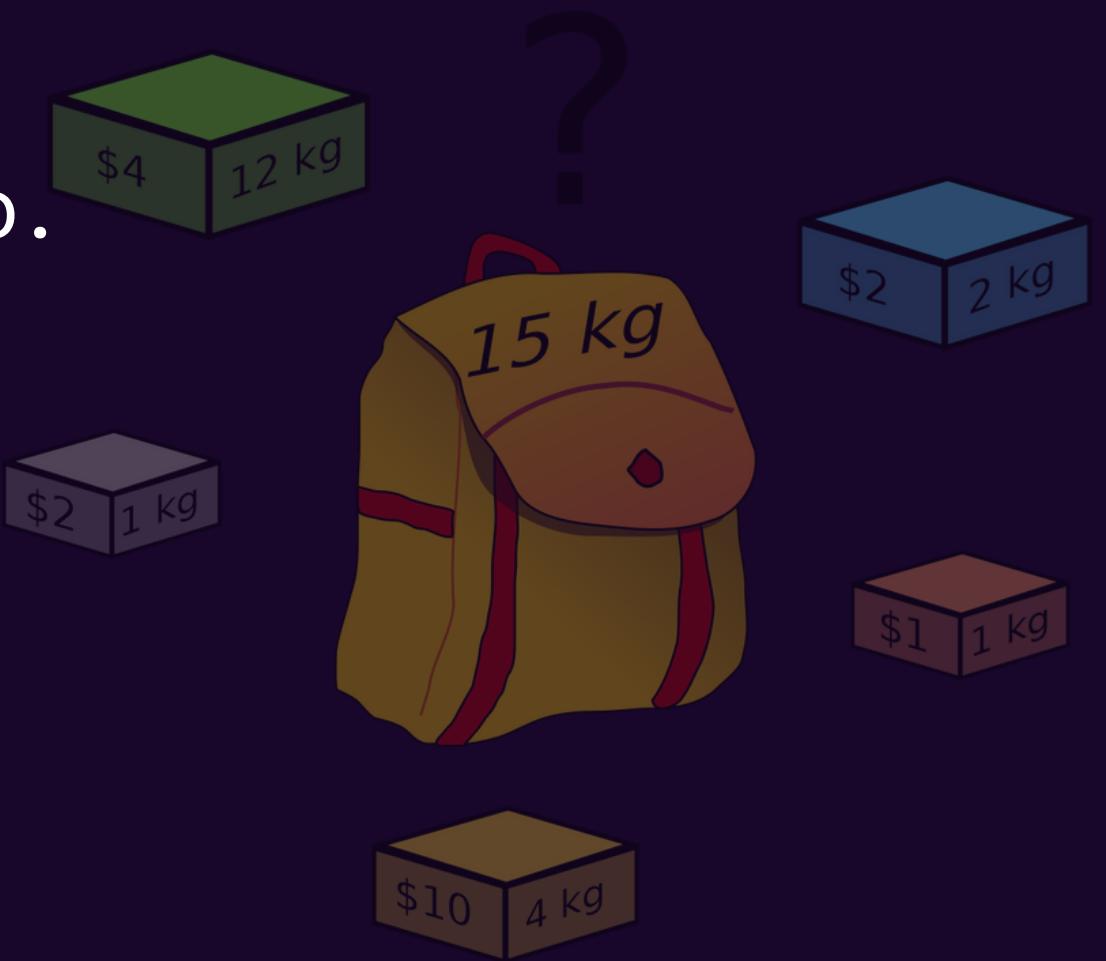
DADOS:

Uma mochila que admite um certo peso;

Um conjunto de objetos, cada um com um valor e um peso.

OBJETIVO:

Selecionar o conjunto de objetos que caibam dentro da mochila de forma a maximizar o valor total dentro da mochila.



PROBLEMA DA MOCHILA BINÁRIA

Qual seria a melhor ordenação da entrada?

ordenar pelo valor/peso

A SOLUÇÃO GULOSA SERÁ ÓTIMA?

veremos...

A SOLUÇÃO GULOSA SERÁ ÓTIMA?

10

R\$60,00

20

R\$100,00

30

R\$120,00

50

MOCHILA

A SOLUÇÃO GULOSA SERÁ ÓTIMA?

20 R\$100, 00

30 R\$120, 00

10 R\$60, 00 MOCHILA

50



A SOLUÇÃO GULOSA SERÁ ÓTIMA?

30 R\$120, 00

20	R\$100, 00	50
10	R\$60, 00	MOCHILA

A SOLUÇÃO GULOSA SERÁ ÓTIMA?

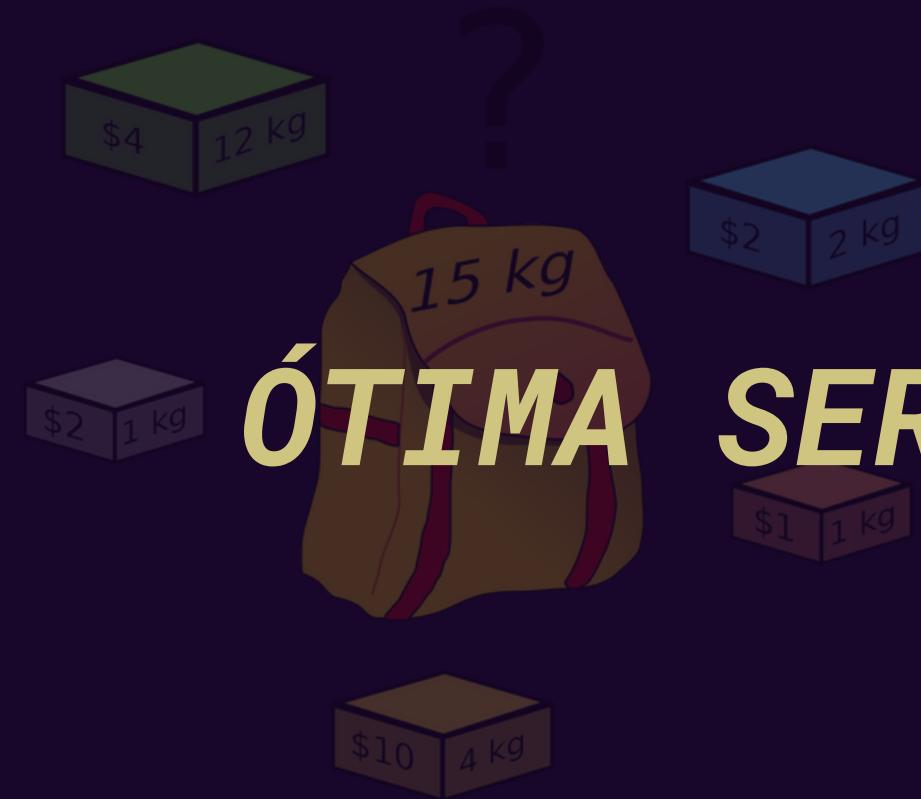
A resposta é

NÃO



TOTAL DE ITENS NA MOCHILA = R\$2.600,00

A SOLUÇÃO GULOSA SERÁ ÓTIMA?



ÓTIMA

SERIA SE...

30

R\$120,00

20

R\$100,00

50

MOCHILA

TOTAL DE ITENS NA MOCHILA = R\$5.600,00

Insertion Sort

- Ordenação simples e eficiente;
- Insertion x Gulosos
 - Conexão com o Processo de Tomada de Decisão;
- Algoritmos Gulosos
 - Decisões Locais;
 - Cada Etapa;
 - Solução Global Ideial;
- Solução Global Ordenada;



Insertion Sort

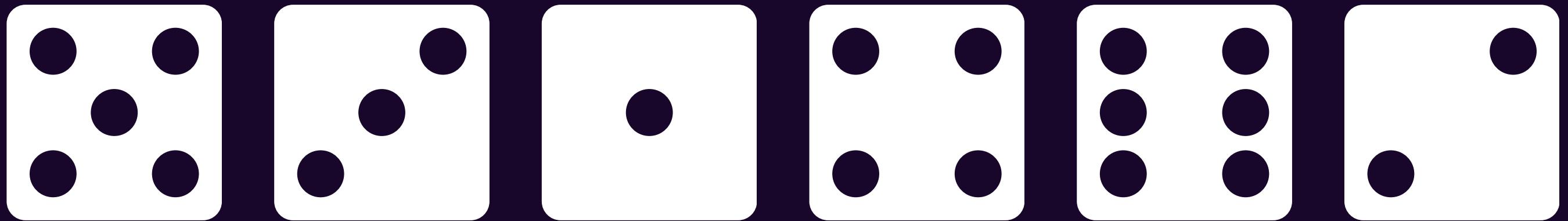
- Processo de Inserção;
- “Pouco Guloso”;
- Sempre garante que a parte já ordenada da lista esteja classificada, em seguida adiciona o próximo elemento da parte não ordenada no lugar certo;



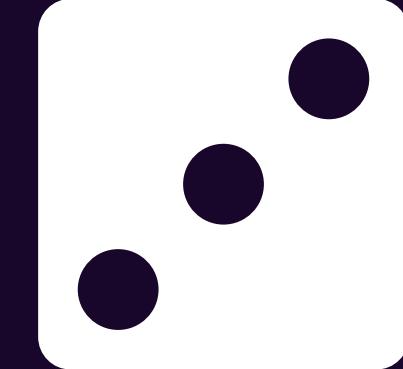
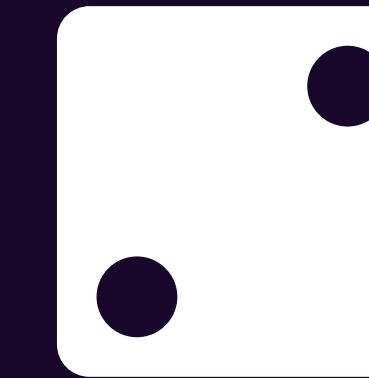
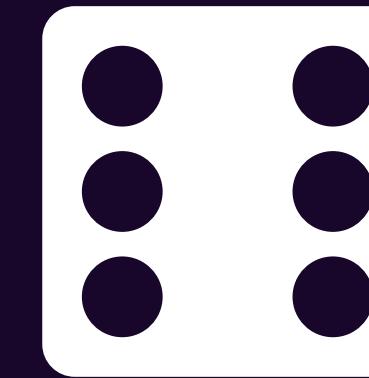
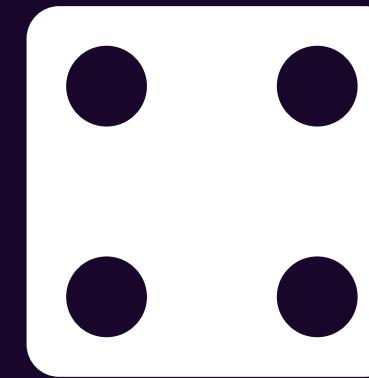
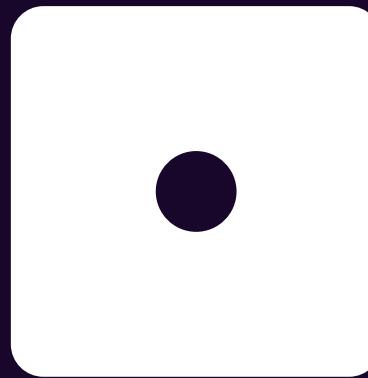
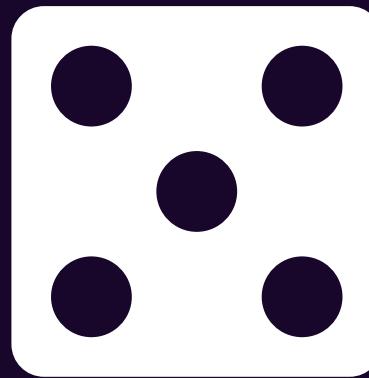
TAYNAN



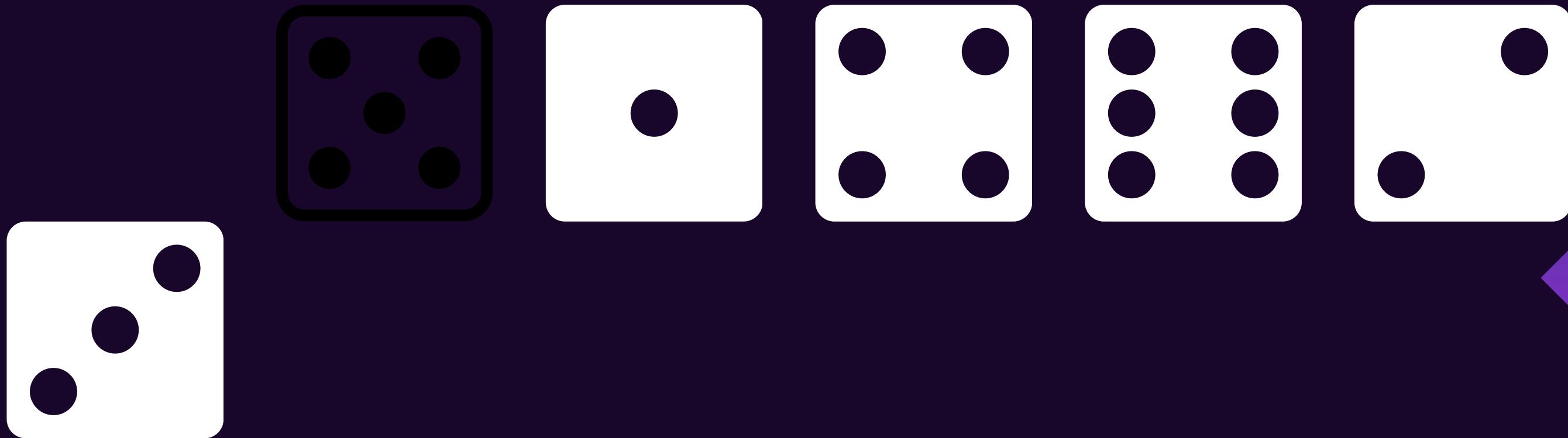
Insertion Sort



Insertion Sort



Insertion Sort

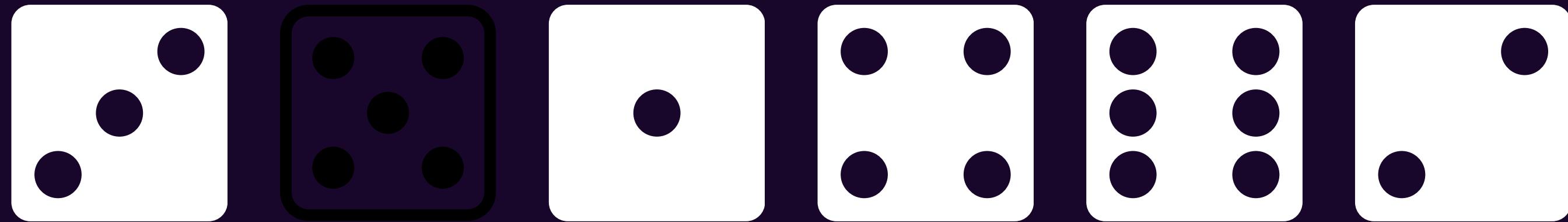




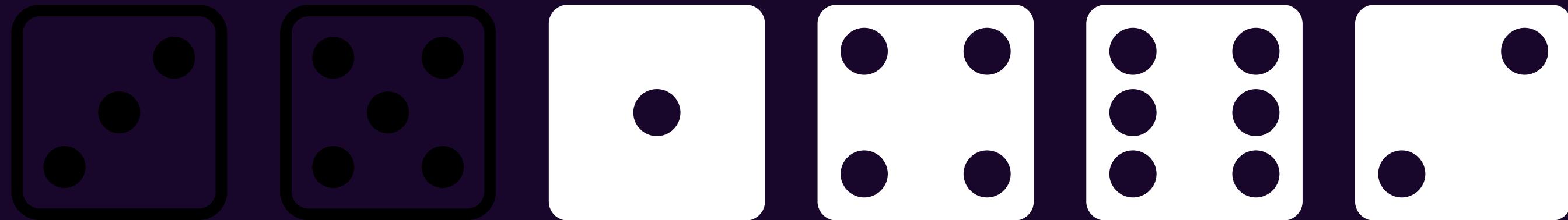
TAYNAN



Insertion Sort

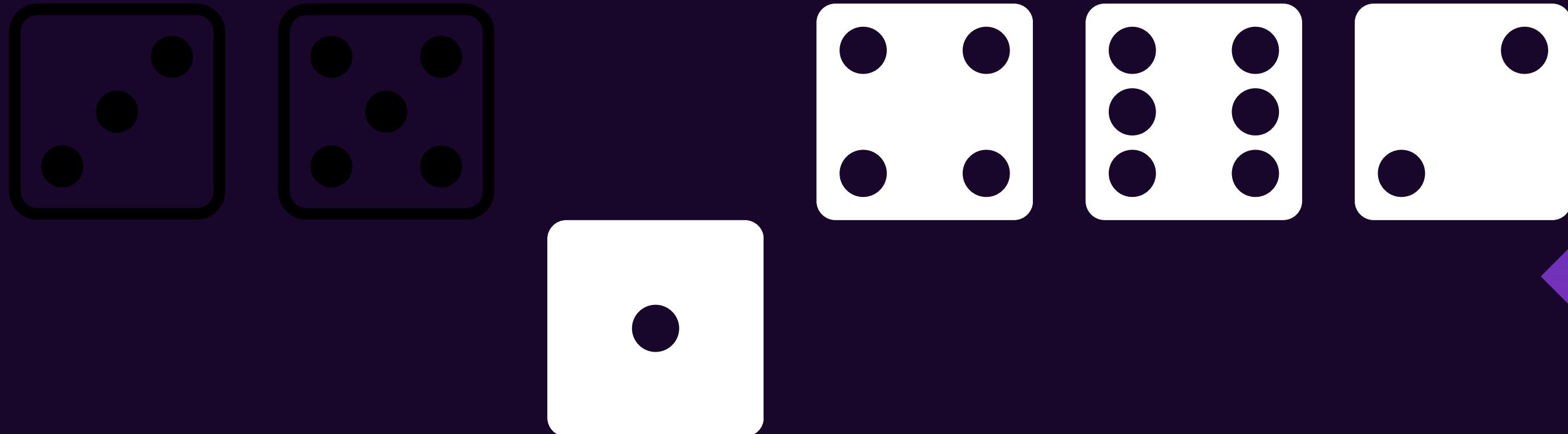


Insertion Sort

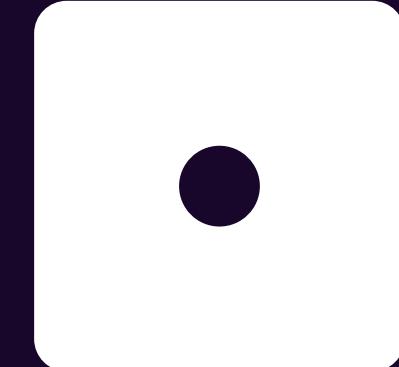
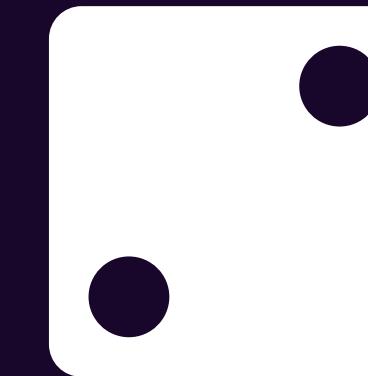
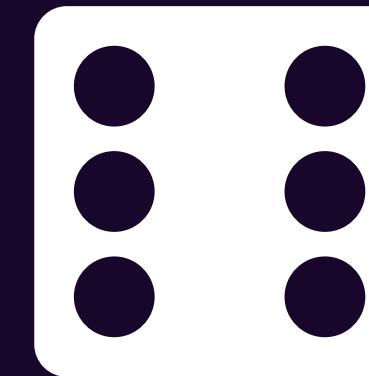
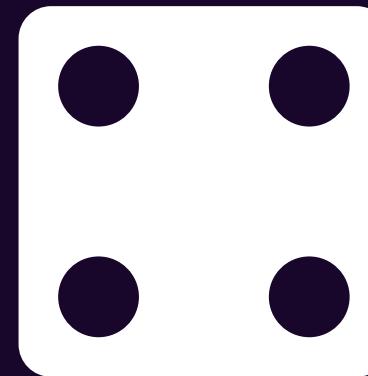
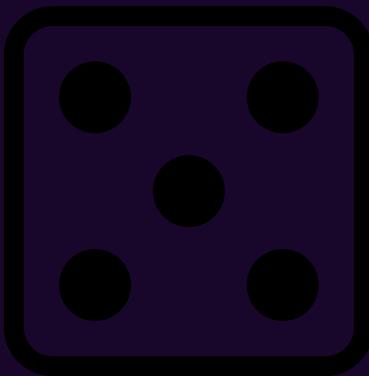




Insertion Sort



Insertion Sort

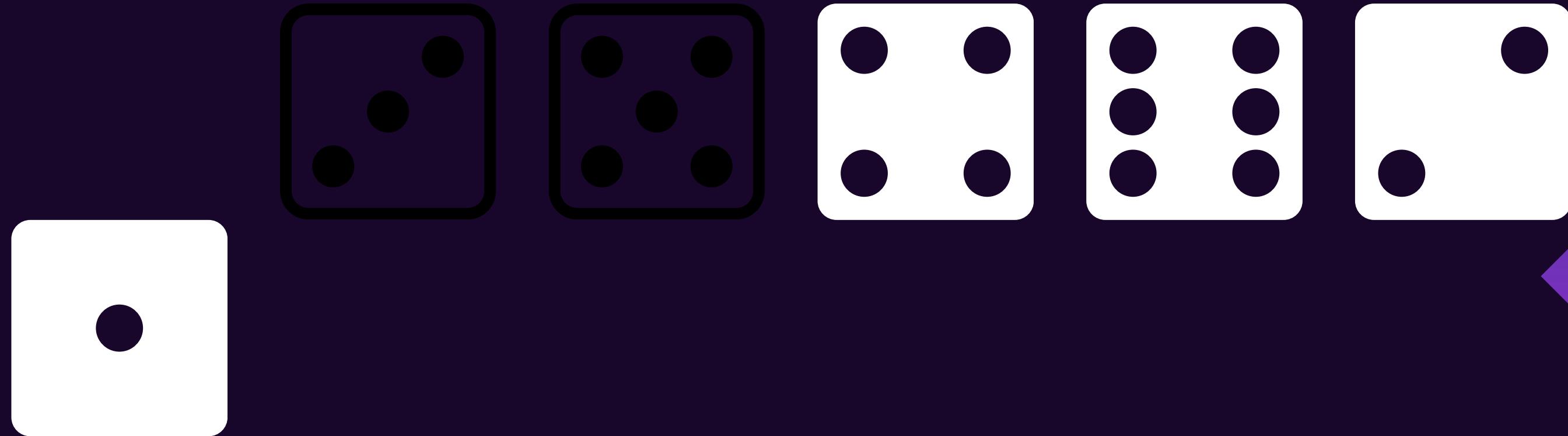




TAYNAN



Insertion Sort

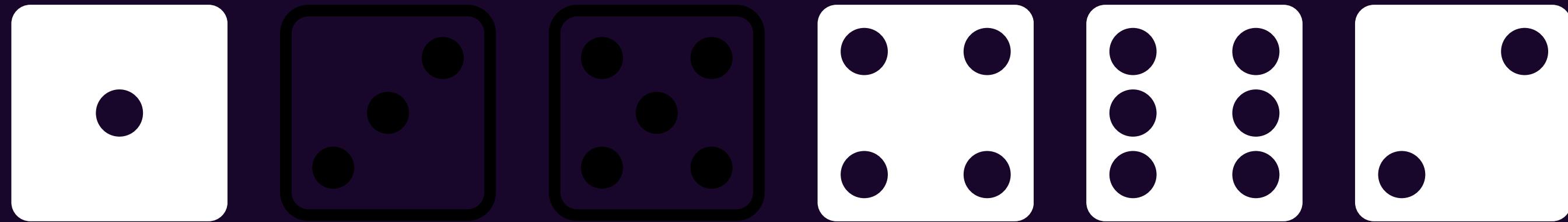




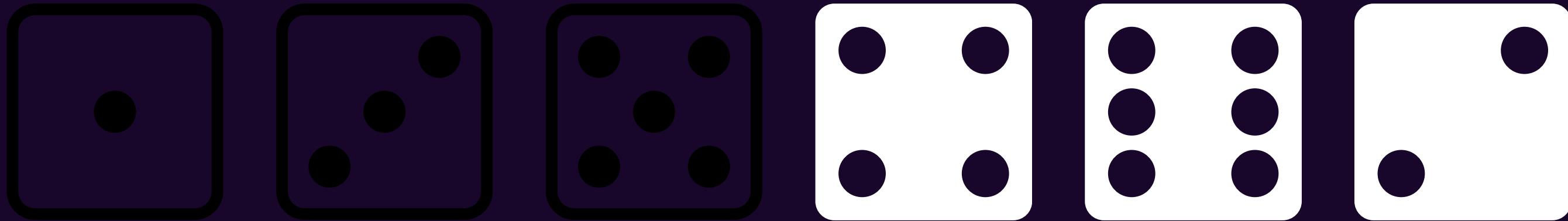
TAYNAN



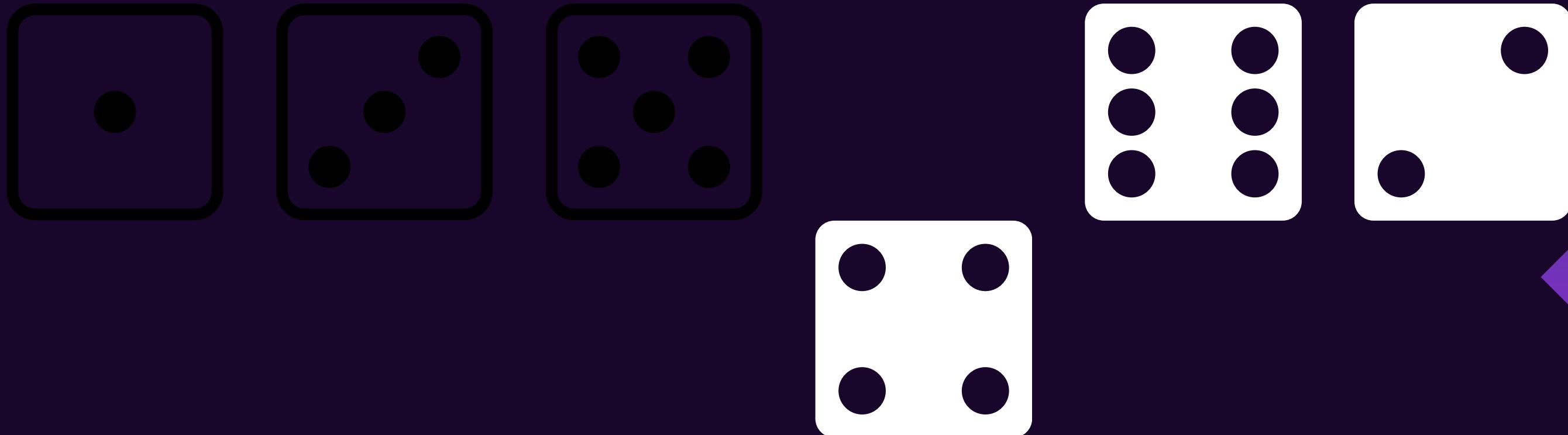
Insertion Sort



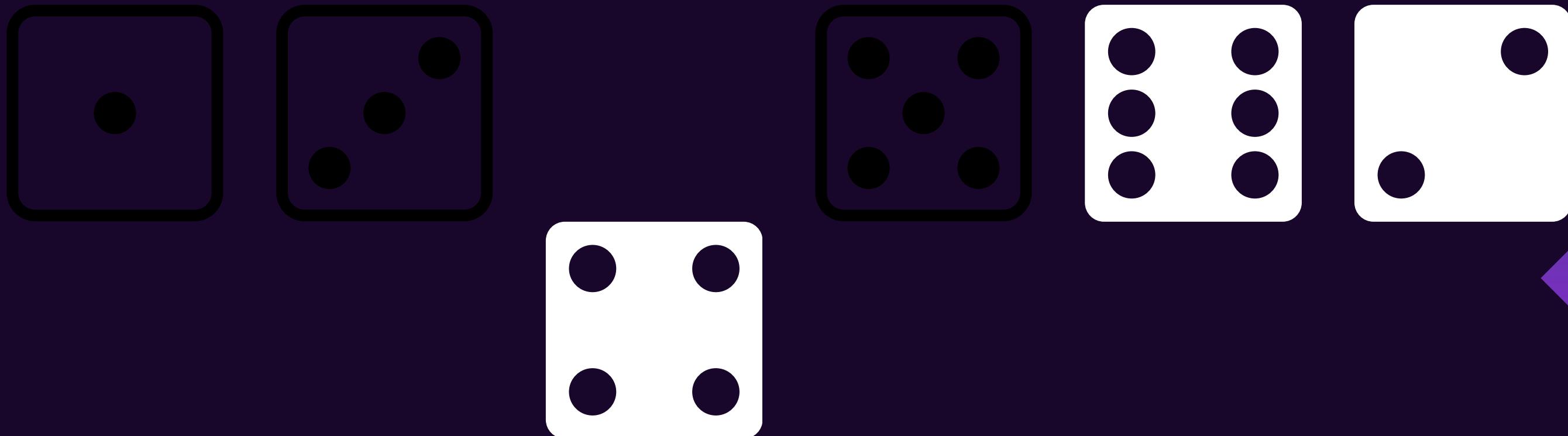
Insertion Sort



Insertion Sort



Insertion Sort

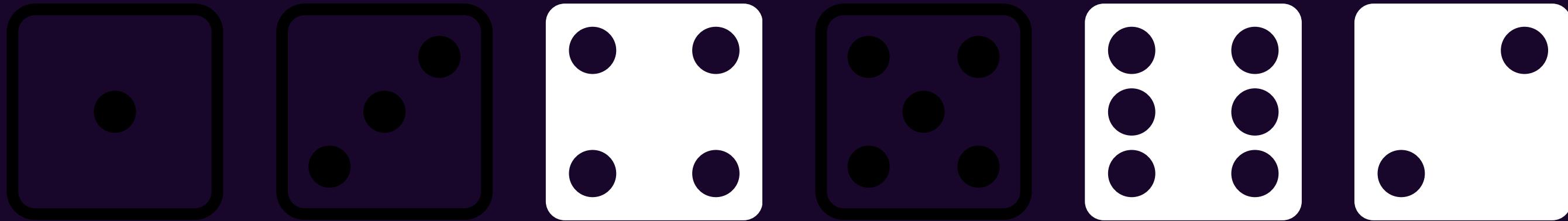




TAYNAN



Insertion Sort

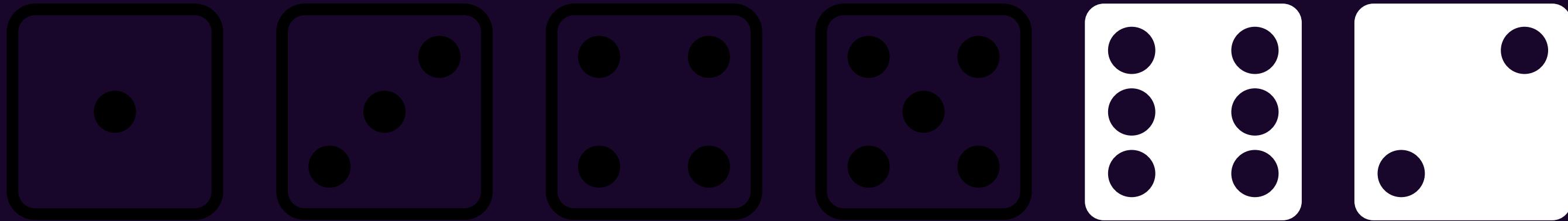




TAYNAN



Insertion Sort

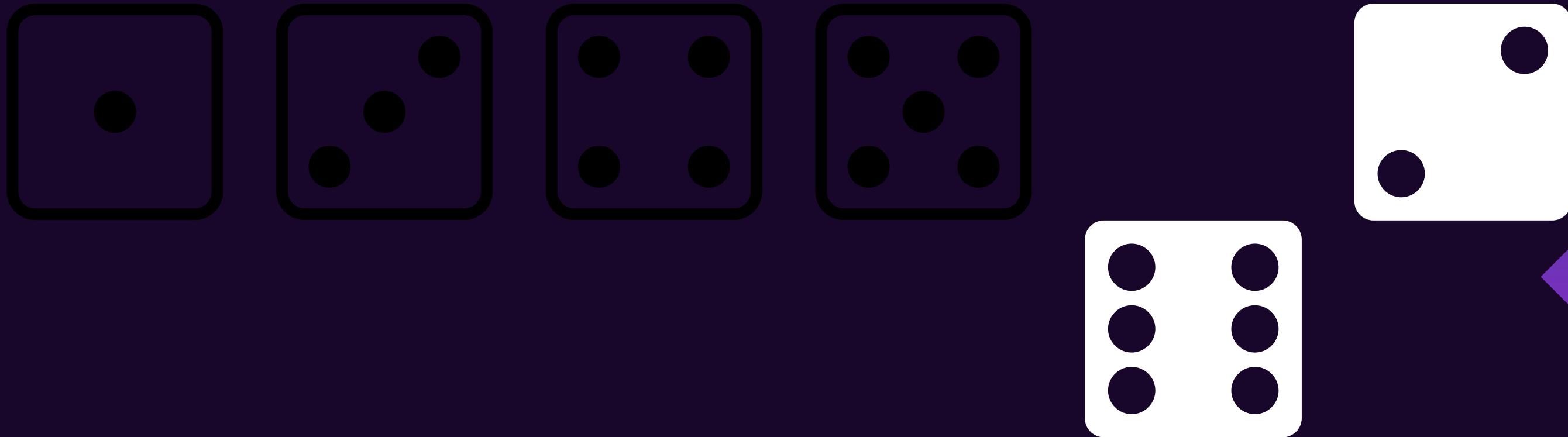




TAYNAN



Insertion Sort

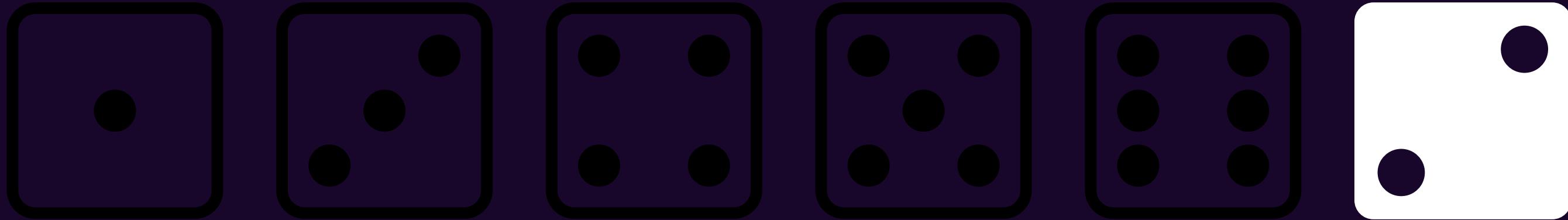




TAYNAN



Insertion Sort

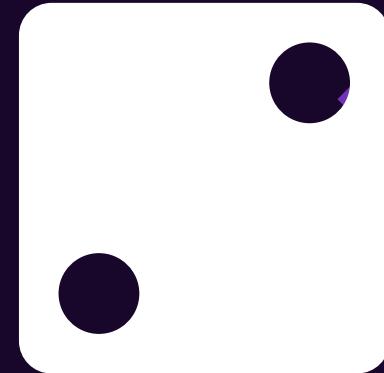
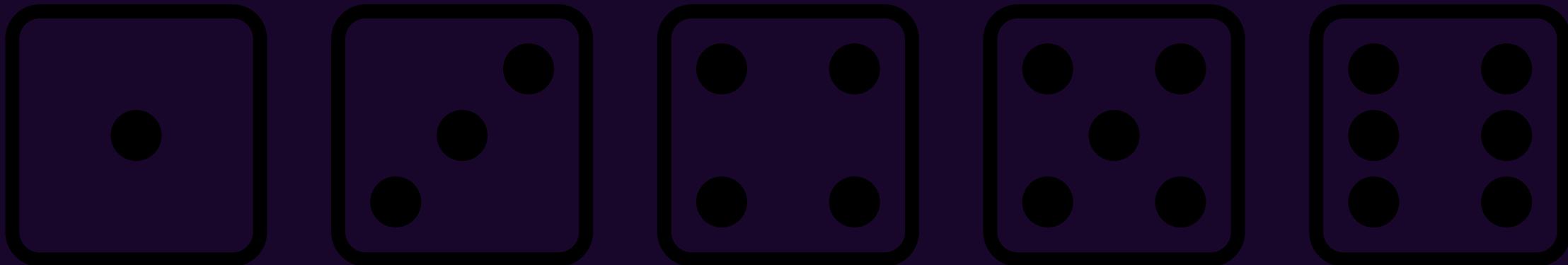




TAYNAN



Insertion Sort

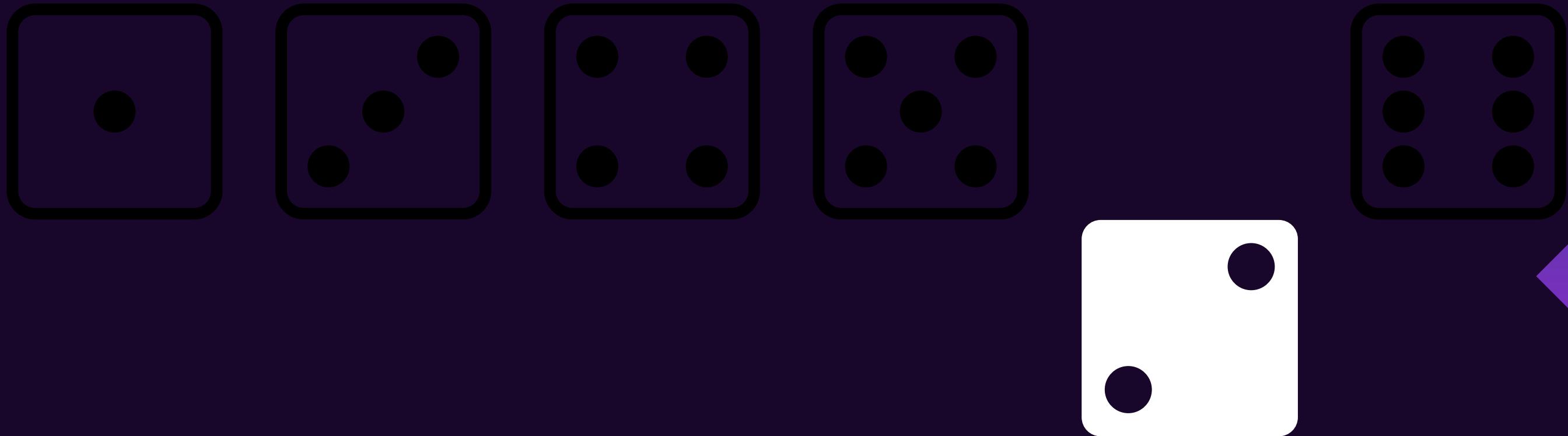




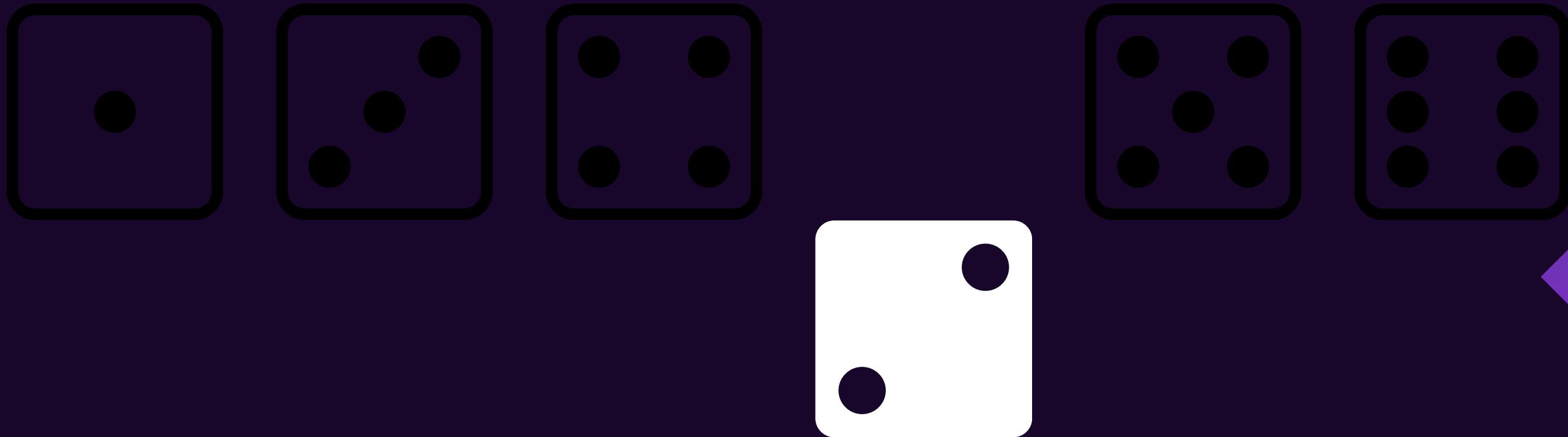
TAYNAN



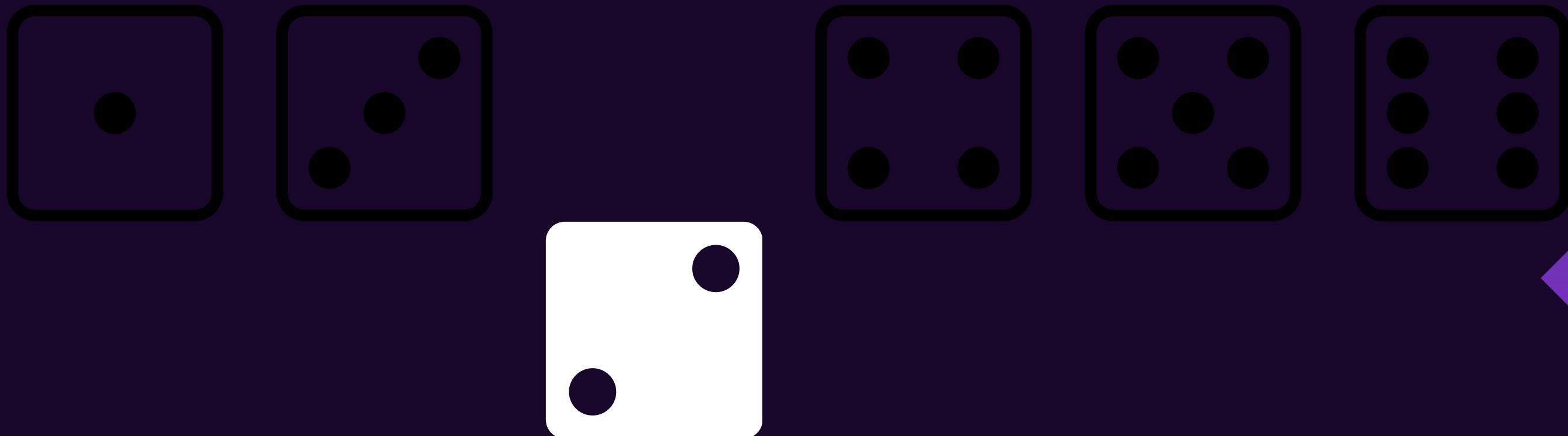
Insertion Sort



Insertion Sort



Insertion Sort

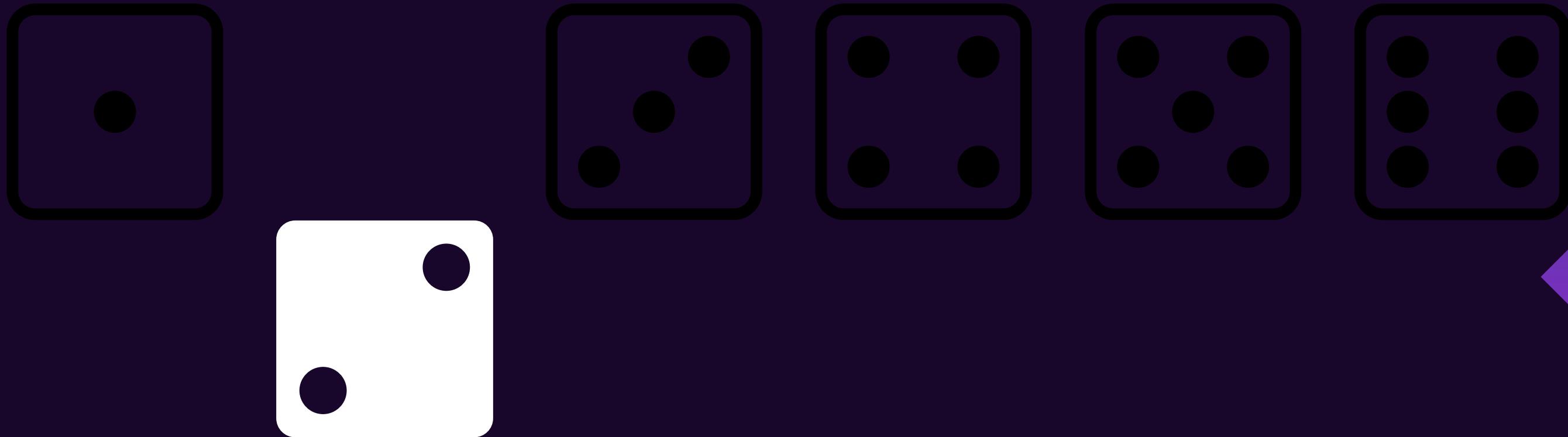




TAYNAN



Insertion Sort

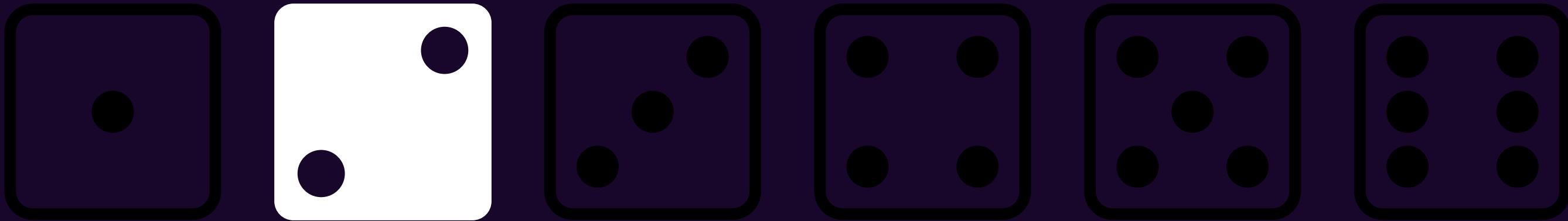




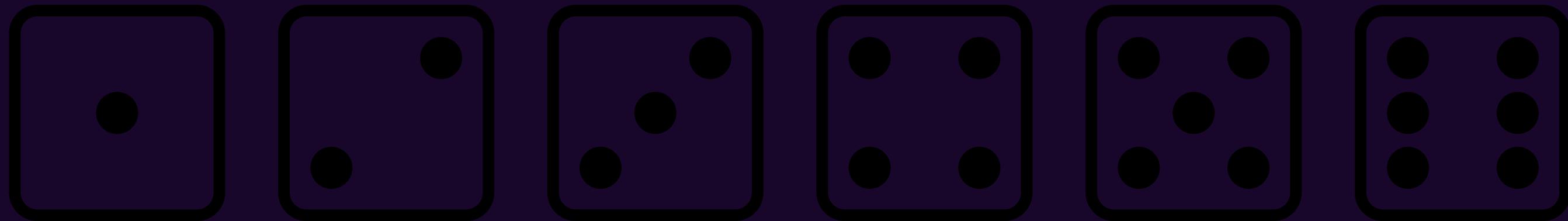
TAYNAN



Insertion Sort



Insertion Sort



Insertion Sort

O ALGORITMO DE INSERÇÃO É "GULOSO" NA MEDIDA EM QUE TOMA A DECISÃO DE INSERIR UM ELEMENTO NA POSIÇÃO CORRETA PARA MANTER A PARTE ORDENADA SEMPRE CORRETA. NO ENTANTO, A SUA ABORDAGEM LOCAL PODE NÃO SER A MELHOR PARA TODOS OS CASOS, JÁ QUE PODE SER MENOS EFICIENTE EM COMPARAÇÃO COM ALGORITMOS MAIS COMPLEXOS EM CERTOS CONJUNTOS DE DADOS.

APESAR DISSO, O INSERTION SORT É ÚTIL PARA LISTAS PEQUENAS OU QUASE ORDENADAS, E SUA LÓGICA DE "INSERIR NO LUGAR CERTO" SE ALINHA DE CERTA FORMA COM O PENSAMENTO DE ALGORITMOS GULOSOS, FOCANDO EM DECISÕES LOCAIS PARA ALCANÇAR UMA SOLUÇÃO GLOBAL ORDENADA.



Vantagens

- Simples e fácil implementação;
- Rápida implementação;
- Pode fornecer a melhor opção (estado atual);
- Quando garantem soluções ótimas, geralmente são os algoritmos mais simples e eficiente;

Desvantagens

- Situacional, só resolve problemas específicos;
- Pode entrar em loop infinito;
- Decisões são irrevogáveis;
- Pode efetuar cálculos repetitivos;
- escolhe caminhos que, à primeira vista, seria mais econômico;

Exemplo Prática

- Descrição de um problema específico de ordenação
- Solução utilizando um algoritmo guloso, como um exemplo customizado de ordenação





Exemplo prático

Você é o proprietário de uma rede de postos de gasolina e recebe uma lista com preços e estoques de fornecedores. A lista começa com o número de fornecedores e a demanda do mercado. Cada fornecedor tem um preço e um estoque de gasolina. Seu objetivo é comprar a quantidade exata exigida pelo mercado ao menor custo possível. Você precisa criar um programa que calcule o gasto mínimo para atender à demanda ou informar se não há estoque suficiente para a compra.



Exemplo prático

1. Ordenar os fornecedores pelos preços do litro de gasolina, do menor para o maior.
2. Comprar a quantidade máxima possível do fornecedor mais barato até atender completamente à demanda do mercado.
3. Se houver sobra de gasolina em fornecedores mais baratos, isso garantirá o menor custo final da compra.
4. A ordenação dos fornecedores pelo preço da gasolina tem complexidade de $O(n * \log n)$.
5. Percorrer o vetor de fornecedores para compra tem complexidade de $O(n)$.
6. Portanto, a complexidade total do algoritmo é $O(n * \log n)$ para resolver o problema.



exemplo prático

algoritmo

```
STRUCT GAS{ DOUBLE PRECO, ESTOQ; };
BOOL COMPARA(GAS X, GAS Y){ RETURN X.PRECO < Y.PRECO; }
GAS FORN[MAXN];
INT N;
DOUBLE D, CUSTO;

INT MAIN(){ SCANF("%D %LF", &N, &D);
FOR(INT I=1; I<=N; I++) SCANF("%LF %LF", &FORN[I].PRECO,
&FORN[I].ESTOQ);

SORT(FORN+1, FORN+N+1, COMPARA);

FOR(INT I=1; I<=N; I++){
GAS DAVEZ=FORN[I];
IF(DAVEZ.ESTOQ<D){
CUSTO+=DAVEZ.ESTOQ*DAVEZ.PRECO;
D-=DAVEZ.ESTOQ; }
ELSE{ CUSTO+=D*DAVEZ.PRECO; D=0; BREAK; }
IF(D) PRINTF("IMPOSSIVEL\n");
ELSE PRINTF("%.2LF\n", CUSTO);

RETURN 0;
}
```

explicando

STRUCT GAS, GUARDA PRECO E ESTOQ => REPRESENTANDO UM FORCEDOR

ARRAY DE GAS NOME DE FORN PARA REPRESENTAR A LISTA DE FORNECEDORES

O ARRAY É ORDENADO COM O SORT E A FUNÇÃO COMPRA, FAZ A COMPARAÇÃO ENTRE DOIS GAS POR VALOR DE PRECO

AO PERCORRER O VETOR VAI ESTAR COMPARANDO SEMPRE A GASOLINA MAIS BARATA E GUARDANDO O VALOR NA VÁRIAVEL CUSTO



TODES



AGRADECIMENTO

APRESENTAÇÃO LÁ ELE



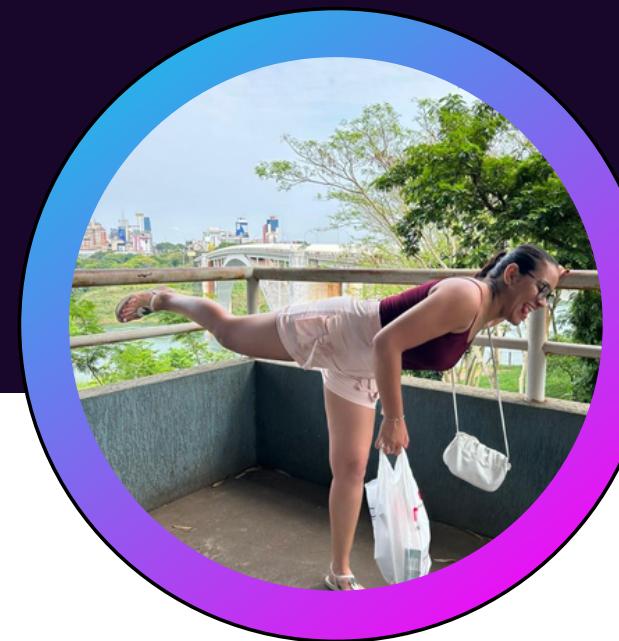
Pink

Empresario
Instagramavel



*Borjão Sem
Fronteiras*

Fã Nautica



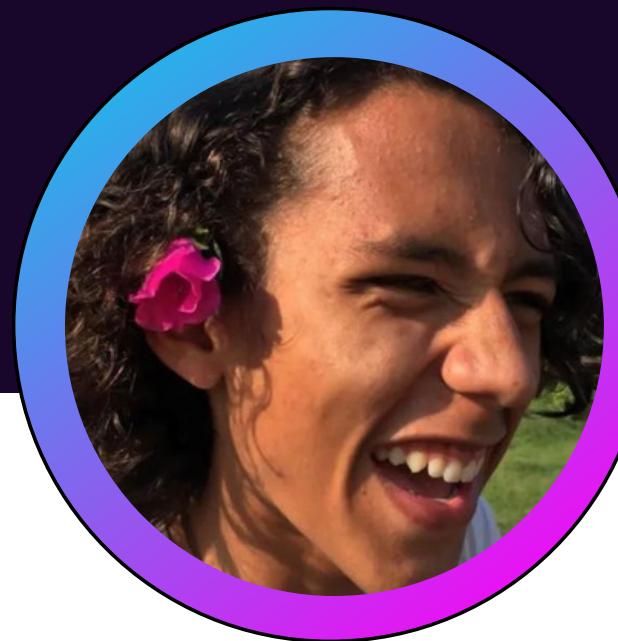
Jaburu Vei Doido

PCD mimado



Marião

Coroinha III



taynoia

DP no
Proerd