INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 07, July 2024, pp: 1200-1208

e-ISSN : 2583-1062

Impact Factor: 5.725

SHUFFLING OF CARDS WITH PYTHON: A COMPREHENSIVE GUIDE

Dr S. Suriya[1], Danish[2]

[1]Associate Professor, Department of Computer Science and Engineering, PSG College of Technology, Coimbatore, Tamil Nadu, India.

[2]PG Scholar, Department of Computer Science and Engineering, PSG College of Technology,Coimbatore, Tamil Nadu, India.

DOI: https://www.doi.org/10.58257/IJPREMS35545

ABSTRACT

The Fisher-Yates Shuffle Algorithm, sometimes referred to as the Knuth Shuffle, is a basic and frequently used technique for varying the element order in an array. This study explores the nuances of the Fisher-Yates Shuffle, offering a thorough analysis of its use and clarifying the fundamental ideas that underpin its effectiveness. The technique is widely used in many different applications, from statistical sampling to cryptographic systems, due to its ease of use and effectiveness. To determine whether the method is appropriate for various use cases, this paper examines the program's mathematical underpinnings and evaluates its time and space complexity.

Keywords: Fisher yates shuffle , Crytographic , Knuth Shuffle.

1. INTRODUCTION

A. Fisher yates - shuffle

The Fisher-Yates shuffle algorithm, also known as the Knuth or Durstenfeld shuffle, is a widely used algorithm for shuffling the elements of an array or a list in a random order. The primary purpose of this algorithm is to generate a random permutation of a collection, meaning that every element in the collection has an equal probability of ending up in any position. The Fisher-Yates shuffle is an in-place algorithm, meaning it shuffles the elements within the existing array or list, making it space-efficient. Here are some key characteristics that a problem should have to be solved effectively with the Fisher-Yates shuffle algorithm:

a) Permutation Requirement
b) Finite Collection
c) In-Place Shuffling
d) Equal Probability
e) Randomness Requirement

The key idea behind the Fisher-Yates shuffle is that at each step, you swap the current element with a randomly selected element that hasn't been processed yet. This guarantees a uniformly random permutation of the elements in the array.

The Fisher-Yates shuffle is a simple and efficient way to shuffle an array or list with a time complexity of O(n) and is widely used in applications where randomness or permutation is required, such as card games, random sampling, and many other algorithms and simulations.

Let's manually trace the Fisher-Yates shuffle algorithm using a deck of UNO cards as an example. Here's a simplified deck of UNO cards with numbers from 0 to 7 for demonstration:

Deck: [0, 1, 2, 3, 4, 5, 6, 7]

We'll perform the Fisher-Yates shuffle step by step:

1. Start with the last card (7) and work your way down to the first card (0).
2. Randomly select a card from the remaining cards and swap it with the card at the current position (from 7 down to 1).
3. Reduce the current position after each swap.

Let's go through the shuffle process:

Initial Deck: [0, 1, 2, 3, 4, 5, 6, 7]

Step 1:

- Current position: 7
- Randomly select a card from the remaining deck (index 2): 2
- Swap card 7 with card 2

Deck: [0, 1, 7, 3, 4, 5, 6, 2]

Step 2:

- Current position: 6
- Randomly select a card from the remaining deck (index 4): 4
- Swap card 6 with card 4

Deck: [0, 1, 7, 3, 6, 5, 4, 2]

Step 3:

- Current position: 5
- Randomly select a card from the remaining deck (index 0): 0
- Swap card 5 with card 0

Deck: [5, 1, 7, 3, 6, 0, 4, 2]

Step 4:

- Current position: 4
- Randomly select a card from the remaining deck (index 1): 1
- Swap card 4 with card 1

Deck: [5, 4, 7, 3, 6, 0, 1, 2]

Step 5:

- Current position: 3
- Randomly select a card from the remaining deck (index 1): 3
- Swap card 3 with card 7

Deck: [5, 4, 3, 7, 6, 0, 1, 2]

Step 6:

- Current position: 2
- Randomly select a card from the remaining deck (index 2): 1
- Swap card 2 with card 1

Deck: [5, 4, 3, 7, 6, 0, 2, 1]

Step 7:

- Current position: 1
- Randomly select a card from the remaining deck (index 0): 5
- Swap card 1 with card 5

Deck: [5, 4, 3, 7, 6, 5, 2, 0]

Step 8:

- Current position: 0 (done)

The deck is now shuffled:

Shuffled Deck: [5, 4, 3, 7, 6, 5, 2, 0]

This completes the Fisher-Yates shuffle process, and your UNO cards are now randomly ordered.

2. LITERATURE SURVEY

In paper [1], proves Without sacrificing the benefits of hyperchaotic systems, satellite image encryption, fisher-yates, 2D filtering, and chaotic mapping can outperform other image encryption algorithms in terms of security.

In paper [2], The algorithm is advised for enhancement in appearance, features, and system foundation in multimedia development life cycle since it efficiently randomizes questions, enhancing individuality.

Paper [3], Fisher Yates prototypes and children's educational games enable speedy design assessments and a variety of question kinds, which improves client engagement and the efficacy of the learning process.

In paper [4], Users can learn Muay Thai with the help of virtual based learning, showcasing the potential of image processing as a substitute teaching method.

Paper [5], Students can improve their mathematical knowledge by using Aritmatika, GDLC, and Fisher Yates. The use of random values in Fisher Yates' work is proven to facilitate learning without the need for copying, according to UAT.

In paper [6], Utilizing the Unity game engine, the application was implemented as a training module that covered fish yates, logistics, and the technology acceptance model. It had a 78.33% perceived rate and a 67.5% acceptance rate.

Paper [7], The puzzle game uses a backtracking algorithm to determine the quickest way to complete a level and is compatible with smartphones that support GDLC.

In paper [8], Steganography offers simplicity, efficiency, and an increased level of security by combining encryption, least significant bit (LSB), randomization, and random keys.

In paper [9], Using digital martial arts, virtual reality, and the Sattolo Shuffle Algorithm, this study investigates Muay Thai movements and shows how useful these methods are for comprehending fundamental Muay Thai movements.

In paper [10], Ludologists use data structures and design patterns to categorize and analyse games. They also test the limits of the models by putting other games in the same engine and trying to provide future applications.

Paper [11], In terms of computational overhead and page swaps, obvious sorting algorithms reach asymptotic optimality, decreasing the likelihood of a fault and the quantity of page swaps needed to retrieve data from insecure memory.

In paper [12], Using metrics and comparisons with current technologies, the IES model—which was created for image encryption, shuffle, confusion, diffusion, and chaotic cryptography—is useful for encrypting large data sets on parallel machines.

In paper [13], Although they demonstrate resilience and security, the zero-watermarking framework, Arnold transformation, phase transformation, RSA, bit error rate, and normalized correlation encounter difficulties in multimedia and document applications.

Paper [14], To improve image transmission security and provide sufficient security and performance, the text presents an image encryption algorithm based on double permutation and random diffusion.

In paper [15], The mobile application, which was created with the Unity game engine and C#, has a 90% satisfaction rating, random quiz distribution, fraction, geometry, and fisher yates' features.

Paper [16], The Fisher-Yates algorithm produces random permutations that result in a robust, two-dimensional map that is resistant to statistical and differential attacks, whereas the cryptosystem algorithm, which is based on digital filters, creates a 2D chaotic map.

In paper [17], On masked and shuffled implementations of Kyber and Saber NIST competitions, the study investigates public key cryptography, post-quantum cryptography, kyber saber, side channel attack, power analysis, and cyberattacks.

Paper [18], By minimizing global memory transactions, computing techniques like bijective shuffle and massively parallel algorithms maximize global memory bandwidth in an effective, deterministic, and labour-efficient manner.

In paper [19], With the use of wavelet transform, 3D shuffling scrambling, and dynamic chaos library, the multi-image encryption algorithm has excellent anti-attack capabilities, optimal ciphertext statistical characteristics, and a high running speed.

Paper [20], Secure communication over existing internet computers and future quantum interest can be built using quantum permutation gates, encryption, decryption, circuits, Qiskit, symmetric cryptography, QKD, QPP, and quantum communication.

In paper [21], Through the process of recovering the secret key from a masked and shuffled implementation of the recently chosen public-key encryption algorithm, the approach to cryptography, post-quantum cryptography, and CRYSTALS-Kyber are demonstrated.

Paper [22], Fault injection, side-channel attacks, ML-KEM, CRYSTALS-Kyber, and post-quantum cryptography are all examined in the study. To improve message recovery success rates and facilitate the recovery of a shared key from a masked and shuffled implementation, it presents a partial key enumeration technique.

Paper [23], Fisher-Yates Shuffle Five Aspect Quality Random Exam Test User Acceptance Test (UAT) has been used to simplify and redesign the English Proficiency Test (EPT). The new system aims to lessen dishonest behaviour and cheating on English training tests by using a website and the Fisher-Yates Shuffle method for randomization.

In paper [24], The Fisher-Yates Shuffle algorithm is used by the online standard test system, HOTS, to reduce cheating. Students are guaranteed a series of multiple-choice questions by means of the system's use of randomization in question-and-answer generation. Platform-based software development makes it simple to integrate and develop with databases, open-source programming languages, and network infrastructure.

In paper [25], Using white box and black box testing, the study compared several randomization techniques, such as the Fisher-Yates Shuffle, learning multimedia, linear congruent method, and Nahwu. The findings demonstrated that the FYS and LCM algorithms are easily implementable due to their similar complexity.

In paper [26], The online questionnaire and algorithm enhancement were conducted to improve the Planned Random algorithm, resulting in a 22.22% decrease in average and total repetitions per simulated case, demonstrating a significant improvement in data randomness.

Paper [27], The study investigates encrypted IoT data manipulation using n-qubit permutation matrices, observing entropy expansion to over 7.9 bits per byte and proposing an entropy expansion algorithm.

In paper [28], With Ed Quizy's anti-cheating features, students are unable to take screenshots or choose random answers, which may improve quiz integrity and deter cheating. Ed Quizy is available for Android devices.

In paper [29], The algorithm performs pixel level and bit-level scrambling, bit-level and pixel-level decomposition, base matrix and coefficient scrambling, and bit-plane and diffusion scrambling in addition to image encryption.

In paper [30], Using chaotic economic maps, researchers investigate image encryption techniques, concentrating on 3D CEM and Fisher-Yates shuffling. Various images are used to test the algorithm, and its security is confirmed by numerical simulations and measurements.

3. PSEUDO CODE – FISHER YATES SHUFFLE

Algorithm: Standard Prim's algorithm

Algorithm 1

```
function fisher_yates_shuffle(arr):
    n = length(arr)  // Get the number of elements in the array.
    for i from n - 1 to 1 do:
        j = random_integer_between (0, i) // Generate a random integer between 0 and i, inclusive.
        swap(arr[i], arr[j]) // Swap the elements at indices i and j.
    end for
end function
function fisher_yates_shuffle(arr):
```

In this pseudo-code:

- **arr** is the array to be shuffled.
- **n** represents the number of elements in the array.
- The loop iterates from **n - 1** down to **1**.
- **random_integer_between(min, max)** generates a random integer between **min** and **max**, inclusive.
- **swap (a, b)** is a function that swaps the elements at indices **a** and **b** in the array.

This algorithm shuffles the elements in the array randomly, ensuring that each element has an equal probability of ending up in any position after the shuffle.

4. APPLICATIONS OF FISHER YATES SHUFFLE ALGORITHM

A. Real-time applications

1. **Randomizing Lists or Arrays:** The primary application of the Fisher-Yates shuffle is to randomly permute the elements of a list or array. This is useful in scenarios where you want to introduce randomness or shuffle the order of items.
2. **Shuffling Cards:** In card games and simulations, the Fisher-Yates shuffle is often used to shuffle decks of cards. This ensures a fair and random distribution of cards before the game starts.
3. **Random Sampling:** When you need to randomly sample a subset of elements from a collection, the Fisher-Yates shuffle can be applied to shuffle the elements and then select the desired number of items.
4. **Quiz or Exam Questions:** In educational settings, especially in computer-based assessments, the Fisher-Yates shuffle can be used to randomize the order of questions or answer choices to prevent bias and cheating.
5. **Playlist Randomization:** Music or media player applications can use the Fisher-Yates shuffle to randomize the order of songs in a playlist, providing users with a varied and unpredictable listening experience.
6. **Randomizing Slides in Presentations**: In presentation software, the algorithm can be used to randomize the order of slides, making each presentation unique and preventing predictability.
7. **Monte Carlo Simulations**: In statistical simulations, the Fisher-Yates shuffle can be employed to generate random permutations, which is useful in Monte Carlo simulations and other statistical modelling techniques.
8. **Password Generation:** While not recommended for security-critical applications, the Fisher-Yates shuffle can be used to generate random passwords by shuffling characters or elements in a set.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 07, July 2024, pp: 1200-1208

9. **Genetic Algorithms:** In genetic algorithms and optimization problems, the Fisher-Yates shuffle may be used to introduce genetic diversity by shuffling the order of elements in the population.
10. **Machine Learning**: In machine learning, the Fisher-Yates shuffle is often used to randomize the order of training data to prevent the model from learning patterns based on the order of input samples.

B. Research applications

1. **Randomized Controlled Trials (RCTs)**: In medical or social science research, randomization is crucial to ensure unbiased treatment assignment. The Fisher-Yates shuffle can be used to randomize the order in which participants receive treatments.
2. **Experimental Design:** When designing experiments, the Fisher-Yates shuffle can be employed to randomize the order of experimental conditions or stimuli presentations to control for order effects and reduce confounding variables.
3. **Randomizing Survey Questions:** In survey research, question order can impact responses. The Fisher-Yates shuffle can be used to randomize the order of survey questions to minimize order bias.
4. **Permutation Testing:** In statistical hypothesis testing, permutation tests involve randomly permuting the data to obtain a distribution under the null hypothesis. The Fisher-Yates shuffle is instrumental in generating these random permutations.
5. **Bootstrapping:** Bootstrapping is a resampling technique used for estimating the sampling distribution of a statistic. The Fisher-Yates shuffle can be utilized in the resampling process to create bootstrap samples.
6. **Simulating Random Events**: In simulation studies, the Fisher-Yates shuffle can model random events and help researchers study the distribution of outcomes under various conditions.
7. **Network Analysis**: When studying network structures, the Fisher-Yates shuffle can be used to randomize the order of nodes or edges in a network, enabling researchers to assess the significance of observed network properties.
8. **Psychological Experiments**: In cognitive psychology, the Fisher-Yates shuffle can be applied to randomize the presentation order of stimuli in experiments, helping researchers control for order effects.
9. **Genomic Studies:** In genomics, researchers may use the Fisher-Yates shuffle to permute the order of genetic markers or samples to assess the significance of observed patterns.
10. **Simulation Studies in Computer Science**: In computer science research, particularly in algorithm analysis, the Fisher-Yates shuffle can be used in simulation studies to assess the performance of algorithms under various random input scenarios.

5. EXPERIMENTAL RESULTS - PRIM'S ALGORITHM

A. Implementation of fisher Yates Shuffle Algorithm in Python

```
import random

def fisher_yates_shuffle(sequence, inplace=True, random_generator=None):
    if not isinstance(sequence, (list, tuple)):
        raise TypeError("Input sequence must be a list or tuple.")

    shuffled_sequence = list(sequence) if not inplace else sequence

    rng = random_generator if random_generator else random

    n = len(shuffled_sequence)

    if n < 2:
        for i in range(n - 1, 0, -1):
            j = rng.randint(0, i)
            shuffled_sequence[i], shuffled_sequence[j] = shuffled_sequence[j], shuffled_sequence[i]

    if not inplace:
        return shuffled_sequence
import random

def fisher_yates_shuffle(sequence, inplace=True, random_generator=None):
    if not isinstance(sequence, (list, tuple)):
        raise TypeError("Input sequence must be a list or tuple.")

    shuffled_sequence = list(sequence) if not inplace else sequence

    rng = random_generator if random_generator else random

    n = len(shuffled_sequence)

    if n < 2:
        for i in range(n - 1, 0, -1):
```

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 07, July 2024, pp: 1200-1208

Output

```
Original Sequence: [1, 2, 3, 4, 5]
Shuffled Sequence (In-Place): [5, 1, 2, 3, 4]
Shuffled Sequence (Copy): [5, 1, 2, 4, 3]
```

This represents the random generated sequence of numbers using Fisher Yates Shuffle algorithm in Python.

B. Complexity Analysis Fisher Yates Shuffle Algorithm

The Fisher-Yates Shuffle has an O(n) time complexity, where n is the array's element count. The algorithm iterates through each array element precisely once, which accounts for the linear time complexity. It chooses an index at random for each iteration and switches the element at that index for the one that is currently in place. O(n) is the total time complexity because the loop executes n times.

The Fisher-Yates Shuffle has an O (1) space complexity, meaning that it uses space continuously. Since the technique shuffles data in-place, it doesn't need more RAM in proportion to the amount of the input. The index and the temporary variable utilized during the swapping procedure are the only temporary variables that require space. No matter how big the array gets, the space complexity never changes.

C. Visualizing of UNO cards in real time with camera detection:

```python
#main file
import sys
import os
import cv2 as cv
import numpy as np
import matplotlib.pyplot as plt
from data_capturing import collectData
from image_processing import trainModels
from image_processing import getCardData
from keras.preprocessing.image import load_img as load

cardColors = ["r", "g", "b", "y"]
cardNumbers = [0,1,2,3,4,5,6,7,8,9,"d","a","n"]
def getUserOption():
    print("")
    print ("-----------------")
    print ("*** Uno Cards ***")
    print ("-----------------")
    print("[1] Read from file")
    print("[2] Live camera stream")
    print("[3] Data capturing")
    print("[0] Exit")
    option = input("::")
    return option
def getUserReadSubMenuOption():
    print("[1] Select card")
    print("[2] All data")
    print("[9] Return")
    option = input("::")
    return option
def getUserCamIndex():
    print("Please input CAMERA INDEX: 0, 1, 2 etc")
    camIndex = input("::")
    return int(camIndex)
def getUserCard():
    print("PLease input COLOR: r,g,b,y")
    userCard = input("::")
    print("Please input NUMBER: 0,1,2,3,4,5,6,7,8,9,d,a,n")
    userCard += input("::")
    return userCard
```

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor: 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 07, July 2024, pp: 1200-1208

```
def liveStream(camIndex):
    print("Opening camera with index " + str(camIndex) + "...")
    vc = cv.VideoCapture(camIndex)
    vc.set(3, 800)
    vc.set(4, 600)
    i = 0
    while vc.isOpened():
        rval, frame = vc.read()
        if i > 20:
            frame = np.asarray(cv.cvtColor(frame, cv.COLOR_BGR2RGB))
            frame, detectedCard = getCardData(True, frame)
            frame = cv.cvtColor(frame, cv.COLOR_RGB2BGR)
        cv.imshow("frame", frame)
        i+=1
        key = cv.waitKey(1)
        if key == 27: break
    cv.destroyWindow("stream")
    vc.release()
def readFromFile(imgName):
    if os.path.isfile("img/" + imgName + ".jpg"):
        frame = np.asarray(load('img/'+ imgName +'.jpg', target_size=(600,800)))
        print("Card Info: " + str(imgName))
        frame, detectedCard = getCardData(False, frame)
        print(plt.imshow(frame))
        print("Detected as: " + str(detectedCard))
        plt.pause(0.001)
print("")
print("Initializing ML models..")
trainModels()
opt = getUserOption()
while opt != "0":
    if opt == "1":
        readOpt = getUserReadSubMenuOption()
        while readOpt != "0":
            if readOpt == "1":  readFromFile(getUserCard())
            elif readOpt == "2":
                for c in cardColors: [readFromFile(c+str(n)) for n in cardNumbers]
            elif readOpt == "9": break
            elif readOpt == "0": sys.exit("Terminated")
            else: print("Invalid option !")
            readOpt = getUserReadSubMenuOption()
    elif opt == "2": liveStream(getUserCamIndex())
    elif opt == "3": collectData(getUserCamIndex())
    elif opt == "0": sys.exit("Terminated")
    else: print("Invalid option !")
    opt = getUserOption()
```

Output:

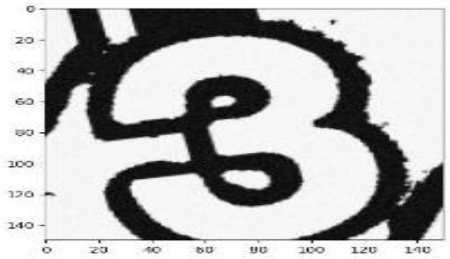

Fig .1

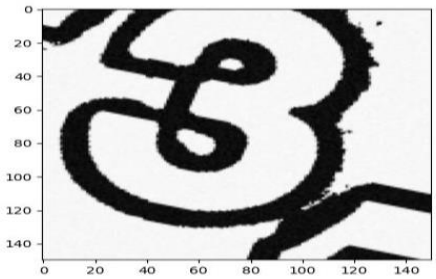

fig.2

in this way we can plot the real time cards in a textual way using real time object detection.

6. CONCLUSION

A well-known and practical algorithm for changing the elemental order in an array is the Fisher-Yates Shuffle, often known as the Knuth Shuffle. This paper delves into the intricacies of the Fisher-Yates Shuffle, shedding light on its numerous applications in domains varying from encryption to statistical sampling. Its simplicity and effectiveness have made it a popular option across many sectors. This paper sheds insight on the Fisher-Yates Shuffle's durability and reliability by providing a comprehensive analysis of its mathematical foundations. The Fisher-Yates Shuffle is a practical method that may be used in a range of situations to rearrange the components in an array, based on the study's conclusions. It is a desirable option for situations where randomness and unpredictability are essential because to its simplicity of implementation and flexibility. As a tried-and-true algorithm, the Fisher-Yates Shuffle is still in use today for a variety of purposes, demonstrating its lasting value in the field of computational methods.

than others.

7. REFERENCES

[1] Naim, M., Ali Pacha, A. A new chaotic satellite image encryption algorithm based on a 2D filter and Fisher Yates shuffling. J Supercomputer 79, 17585–17618 (2023). https://doi.org/10.1007/s11227-023-05346-5.

[2] Khoiru Nurfitri, Ismail Abdurrozzaq, Jamilah KaramanPenerapan. Algoritma Fisher Yates Shuffle pada Game Edukasi "English For Children" di LKP Elite English School Ponorogo.https://doi.org/10.47709/digitech.v3i2.2885

[3] Lian Sholahudin,Iwan Giri Waluyo. PENERAPAN ALGORITMA FISHER-YATES DALAM APLIKASI MEDIA PEMBELAJARAN PENGENALAN OBJEK MENGGUNAKAN METODE PROTOTYPE BERBASIS ANDROID

[4] martin wongo , wirawan istiono.Learn Muay Thai Basic Movement in Virtual Reality . http://dx.doi.org/10.46729/ijstm.v4i2.759

[5] Yulyanto,Andriasnyah,Nunu nugraha. Rancang Bangun Game Pembelajaran Operasi Dasar Matematika Menggunakan Algoritma Fisher Yattes .https://doi.org/10.47065/bit.v4i2.704

[6] Jonathan Franzeli 1, Wirawan Istiono* 2. Advancing Logistics Training with Virtual Reality: A Case Study of PT XYZ's Innovative Approach in Indonesia

[7] Yulyanto1, Fikri Maudia Arsyad2 , Tito Sugiharto2.Pengembangan Game Puzzle Find Grass Menggunakan Algoritma Backtracking .https://doi.org/10.47065/bit.v4i2.703

[8] Khaled H. Abuhmaidan1*, Ahmad K. Kayed2, and Maryam Alrisia.Steganography: A Flexible Embedded Randomization Technique

[9] Martin Wongso1*, Wirawan Istiono2.Learn Muay Thai Basic Movement in Virtual Reality and Sattolo Shuffle Algorithm.http://dx.doi.org/10.46729/ijstm.v4i2.759

[10] Matthias Steinböck.Game Design Patterns in Digital Card Games (A browser engine to an abstract card game model)

[11] Tianyao Gu, Yilei Wang, Bingnan Chen. Efficient Oblivious Sorting and Shuffling for Hardware Enclaves.https://ia.cr/2023/1258

[12] Alaa Abdulsalm Alarood a, Eesa Alsolami a , Mahmoud Ahmad Al-Khasawneh b , Nedal Ababneh c , el Elmedany d. Hyper-chaotic plain image encryption scheme using improved shuffled confusion-diffusion . https://doi.org/10.1016/j.asej.2021.09.010

[13] HSIU-CHI TSENG, KING-CHU HUNG. Robust and Secure Zero-Watermark architecture With Arnold and Phase Transformation.https://doi.org/10.1109/ACCESS.2023.3309289

[14] Jiming Zheng, mingkun xue .A Novel Image Encryption Algorithm Based on Double Permutation and Random Diffusion.https://doi.org/10.21203/rs.3.rs-1917006/v1

[15] Andrio Effendi, Wirawan Istiono. FRACTIONS AND GEOMETRY MATH MATERIAL IN EDUCATION APPLICATION.

[16] Z.B. Madouri, N. Hadj Said, A. Ali Pacha. Image encryption algorithm based on digital filters controlled by 2D robust chaotic map.https://doi.org/10.1016/j.ijleo.2022.169382

[17] LINUS BACKLUND. A Side-Channel Attack on Masked and Shuffled Implementations of M-LWE and M-LWR Cryptography (a case study of kyber and saber)

[18] Rony Mitchell, Daniel stokes, eibe frank . Bandwidth-Optimal Random Shuffling for GPU.https://doi.org/10.48550/arXiv.2106.06161

[19] Zhong, H., Li, G. Multi-image encryption algorithm based on wavelet transform and 3D shuffling scrambling. Multimed Tools Appl 81, 24757–24776 (2022). https://doi.org/10.1007/s11042-022-12479-x

[20] Kuang, R., Perepechaenko, M. Quantum encryption with quantum permutation pad in IBMQ systems. EPJ Quantum Technol. 9, 26 (2022). https://doi.org/10.1140/epjqt/s40507-022-00145-y

[21] Backlund, L., Ngo, K., Gärtner, J., Dubrova, E. (2023). Secret Key Recovery Attack on Masked and Shuffled Implementations of CRYSTALS-Kyber and Saber. In: Zhou, J., et al. Applied Cryptography and Network Security Workshops. ACNS 2023. Lecture Notes in Computer Science, vol 13907. Springer, Cham. https://doi.org/10.1007/978-3-031-41181-6_9

[22] Sonke Jendral, Kalle Ngo, Ruize Wang and Elena Dubrova.A Single-Trace Message Recovery Attack on a Masked and Shuffled Implementation of CRYSTALS-Kyber.Paper 2023/1587.https://ia.cr/2023/1587

[23] Andri Wijaya1, Frendy Dodo Chang. Information System Web Design For Class English Proficiency Test Using Fisher Yates Shuffle Method.April 2023bit-Tech 5(3):166-http://dx.doi.org/10.32877/bt.v5i3.724

[24] Mulyahadi Wijaya Duha, Arie Rafika Dewi, Eka Rahayu.Online Standard Test System Based on HOTS Using Fisher-Yates Shuffle Algorithm.https://doi.org/10.24114/cess.v7i2.35657

[25] Ukan saokani , mohammed ifran , rachemet jaenel abibin , icshan taufik .Comparison of the Fisher-Yates Shuffle and the Linear Congruent Algorithm for Randomizing Questions in Nahwu Learning Multimedia.https://doi.org/10.24114/cess.v7i2.35657

[26] Jemuel Fontilaa, Jiro Mark Garciaa , Mark Jimwell Lagartaa , Mark Christopher Blancoa , Dan Michael Corteza.DOI :10.26524/sajet.2022.12.018

[27] Avval Amil , and Shashank Gupta. Quantum entropy expansion using n-qubit permutation matrices in Galois field.https://doi.org/10.48550/arXiv.2207.07148

[28] Wan Osman, W. N. N., & Ab Rahman, T. D. N. H. (2023). EdQuizy: A Development of a Quiz Mobile application with Anti-Cheating Features. Applied Information Technology and Computer Science, 4(1),168–184.Retrievedfrom https://publisher.uthm.edu.my/periodicals/index.php/aitcs/article/view/7553

[29] Wang, S., Sun, B., Wang, Y. et al. Image encryption algorithm using multi-base diffusion and a new four-dimensional chaotic system. Multimed Tools Appl (2023).

[30] Karawia, A. A. (2019). Image encryption based on fisher-yates shuffling and three-dimensional chaotic economic map. IET Image Processing, 13(12), 2086-2097. https://doi.org/10.1049/iet-ipr.2018.5142