ระบบจัดการข้อมูลพนักงาน

นาย อลีฟ แวหะยี 6706022510034

นาย วิสิฐศักดิ์ เพชรหนองชุม 6706022510123

นาย นนทวัฒน์ ประสพรัตน์ 6706022510212

นางสาว วริศรา สุขสวัสดิ์ 6706022510221

โครงงานนี้เป็นส่วนหนึ่งของการศึกษาหลักสูตรวิศวกรรมศาสตรบัณฑิต

สาขาวิชาวิศวกรรมสารสนเทศและเครือข่าย ภาควิชาเทคโนโลยีสารสนเทศคณะเทคโนโลยีและการจัดการอุตสาหกรรม มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

ปีการศึกษา 2567

ลิขสิทธิ์ของมหาลัยมหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ

คำนำ

การจัดทําโครงงาน “ระบบจัดการข้อมูลพนักงาน” นี้เป็นส่วนหนึ่งของวิชา COMPUTER PROGRAMMIMG ของหลักสูตรวิศวกรรมศาสตรบัณฑิต สาขาวิชาวิศวกรรมสารสนเทศและเครือข่ายภาควิชาเทคโนโลยีสารสนเทศคณะเทคโนโลยีและการจัดการอุตสาหกรรม มหาวิทยาลัยเทคโนโลยีพระจอมเกล้าพระนครเหนือ เพื่อให้นักศึกษาได้นําความรู้ที่เรียนมาทั้งหมดมาประยุกต์ใช้ใน  
การพัฒนาโปรแกรมที่สามารถทํางานได้จริง โดยเน้นการออกแบบและเขียนโปรแกรมในภาษา Python ซึ่งเป็นภาษาที่เรียนมาในวิชา COMPUTER PROGRAMMING โดยโครงงานนี้จะช่วย การคิดวิเคราะห์และแก้ปัญหาทางเทคนิค เพื่อเตรียมความพร้อมในการประกอบอาชีพด้านวิศวกรรมสารสนเทศและเครือข่ายในอนาคต หากมีข้อผิดพลาดประการใด คณะผู้จัดทําต้องขออภัยไว้ ณ ที่นี้ด้วย

สารบัญ

หน้า

[คำนำ ข](#_Toc179395340)

[สารบัญ ค](#_Toc179395341)

[สารบัญภาพ ช](#_Toc179395342)

[บทที่ 1 บทนํา 1](#_Toc179395343)

[1.1 วัตถุประสงค์ของโครงการ 1](#_Toc179395344)

[1.2 ขอบเขตของโครงการ 1](#_Toc179395345)

[1.3 ประโยชน์ที่ได้รับ 1](#_Toc179395346)

[1.4 เครื่องมือที่ใช้งาน 1](#_Toc179395347)

[บทที่ 2 การทำงานของระบบจัดการข้อมูลพนักงาน 2](#_Toc179395348)

[2.1 ฟิลด์ในการจัดการข้อมูลพนักงาน 2](#_Toc179395349)

[2.1.1 ID (รหัสพนักงาน) 2](#_Toc179395350)

[2.1.2 Name (ชื่อพนักงาน) 2](#_Toc179395351)

[2.1.3 Department (แผนก) 2](#_Toc179395352)

[2.1.4 Score (คะแนน) 2](#_Toc179395353)

[2.1.5 Salary (เงินเดือน) 2](#_Toc179395354)

[2.2 ฟังก์ชันการทำงานของระบบจัดการพนักงาน 3](#_Toc179395355)

[2.2.1 ฟังก์ชัน main 3](#_Toc179395356)

[2.2.2 ฟังก์ชัน Show all data 3](#_Toc179395357)

[2.2.3 ฟังก์ชัน Insert Data 4](#_Toc179395358)

[2.2.4 ฟังก์ชัน Edit data 5](#_Toc179395359)

[2.2.5 ฟังก์ชัน Delete 9](#_Toc179395360)

[2.2.6 ฟังก์ชัน Export 9](#_Toc179395361)

[2.2.7 ฟังก์ชัน Exit 10](#_Toc179395362)

[บทที่ 3 อธิบายการทำงานของ code ส่วนการประมวลผล 11](#_Toc179395363)

[3.1 การเตรียมความพร้อมก่อนการเขียนโปรแกรม 11](#_Toc179395364)

[3.1.1 การนำเข้าโมดูลและการกำหนดค่าเริ่มต้น 11](#_Toc179395365)

[3.1.2 การกำหนดเส้นทางของไฟล์ไบนารี 11](#_Toc179395366)

สารบัญ (ต่อ)

หน้า

[3.2 การอ่านค่าจากไฟล์ไบนารี 11](#_Toc179395367)

[3.2.1 การประกาศตัวแปรไว้เก็บข้อมูลไปประมวลผล 12](#_Toc179395368)

[3.2.2 การเปิดไฟล์ 12](#_Toc179395369)

[3.2.3 การอ่านข้อมูลก่อนนำไปประมวลผล 12](#_Toc179395370)

[3.2.4 การประมวลผลข้อมูลที่อ่านได้ 13](#_Toc179395371)

[3.2.5 การจัดการข้อผิดพลาด 13](#_Toc179395372)

[3.2.6 ผลลัพธ์ที่ได้จากการเรียกใช้ฟังก์ชัน 13](#_Toc179395373)

[3.2.7 ตัวอย่างการเรียกใช้ 13](#_Toc179395374)

[3.3 การเขียนลงในไฟล์ไบนารี 13](#_Toc179395375)

[3.3.1 การตรวจสอบความยาวของลิสต์ข้อมูล 14](#_Toc179395376)

[3.3.2 การวนลูปเขียนข้อมูล 14](#_Toc179395377)

[3.3.3 การจัดรูปแบบข้อมูลก่อนเขียน 15](#_Toc179395378)

[3.3.4 ผลลัพธ์ที่ได้ 15](#_Toc179395379)

[3.3.5 ตัวอย่างการเรียกใช้ 15](#_Toc179395380)

[3.4 การแก้ไขข้อมูลในไฟล์ไบนารี 15](#_Toc179395381)

[3.4.1 การอ่านข้อมูลก่อนแก้ไขข้อมูล 16](#_Toc179395382)

[3.4.2 การเลือกคอลัมน์ที่จะแก้ไขข้อมูล 16](#_Toc179395383)

[3.4.3 กรณีการแก้ไขตามคอลัมน์ 16](#_Toc179395384)

[3.4.4 การจัดการข้อมูลที่ไม่ถูกต้อง 19](#_Toc179395385)

[3.4.5 การเขียนข้อมูลกลับลงไฟล์ 19](#_Toc179395386)

[3.4.6 ผลลัพธ์ที่ได้ 19](#_Toc179395387)

[3.4.7 ตัวอย่างการเรียกใช้ 19](#_Toc179395388)

สารบัญ (ต่อ)

หน้า

[3.5 การเพิ่มข้อมูลใหม่ลงในไฟล์ไบนารี 20](#_Toc179395389)

[3.5.1 การอ่านข้อมูลจากไฟล์เพื่อตรวจสอบ ID ซ้ำซ้อน 20](#_Toc179395390)

[3.5.2 การตรวจสอบ ID ซ้ำ 20](#_Toc179395391)

[3.5.3 การเพิ่มข้อมูลใหม่ 21](#_Toc179395392)

[3.5.4 การเขียนข้อมูลใหม่ลงไฟล์ 21](#_Toc179395393)

[3.5.5 ผลลัพธ์ที่ได้ 22](#_Toc179395394)

[3.5.6 ตัวอย่างการเรียกใช้ 22](#_Toc179395395)

[3.6 การลบข้อมูลในไฟล์ไบนารี 22](#_Toc179395396)

[3.6.1 การอ่านข้อมูลก่อนลบข้อมูล 22](#_Toc179395397)

[3.6.2 การตรวจสอบ ID ที่จะลบ 23](#_Toc179395398)

[3.6.3 การลบเรคคอร์ด 23](#_Toc179395399)

[3.6.4 การเขียนข้อมูลกลับลงไฟล์ 24](#_Toc179395400)

[3.6.5 การแสดงผล ID ที่ถูกลบ 24](#_Toc179395401)

[3.6.6 ผลลัพธ์ที่ได้ 24](#_Toc179395402)

[3.6.7 ตัวอย่างการเรียกใช้ 24](#_Toc179395403)

[3.7 การสร้างรายงานสรุปข้อมูลและบันทึกเป็นไฟล์ข้อความ 24](#_Toc179395404)

[3.7.1 การอ่านข้อมูลจากไฟล์ 25](#_Toc179395405)

[3.7.2 การจัดกลุ่มตามแผนก 25](#_Toc179395406)

[3.7.3 การหาพนักงานของแผนกที่ดีที่สุด 26](#_Toc179395407)

[3.7.4 การสร้างข้อความรายงาน 27](#_Toc179395408)

[3.7.5 การบันทึกลงไฟล์ 27](#_Toc179395409)

[3.7.6 การแสดงผล 27](#_Toc179395410)

[3.7.7 ผลลัพธ์ที่ได้ 27](#_Toc179395411)

[3.7.8 ตัวอย่างการเรียกใช้ 28](#_Toc179395412)

สารบัญ (ต่อ)

หน้า

[3.8 แสดงข้อมูลของพนักงานทั้งหมด 28](#_Toc179395413)

[3.8.1 การอ่านข้อมูลจากไฟล์ 28](#_Toc179395414)

[3.8.2 การเลือกโหมดการแสดงข้อมูล 29](#_Toc179395415)

[3.8.3 การแสดงข้อมูลในกรณี choice = '1' (แสดงคะแนนเฉลี่ย) 29](#_Toc179395416)

[3.8.4 การแสดงข้อมูลในกรณี choice = '2' (แสดงคะแนนแบบแจกแจ้ง) 30](#_Toc179395417)

[3.8.5 การตรวจสอบและเรียงลำดับข้อมูล 30](#_Toc179395418)

[3.8.6 การตรวจสอบข้อมูล 30](#_Toc179395419)

[3.8.7 ผลลัพธ์ที่ได้ 31](#_Toc179395420)

[3.8.8 ตัวอย่างการเรียกใช้ 31](#_Toc179395421)

[3.9 แสดงข้อมูลเฉพาะของพนักงานตามคอลัมน์ที่กำหนด 31](#_Toc179395422)

[3.9.1 รายละเอียดการทำงาน 33](#_Toc179395423)

[3.9.2 การอ่านข้อมูลจากไฟล์ 33](#_Toc179395424)

[3.9.3 การกำหนดคอลัมน์ที่ต้องการค้นหา 33](#_Toc179395425)

[3.9.4 การจัดการข้อมูลในคอลัมน์ Score 33](#_Toc179395426)

[3.9.5 การจัดการข้อมูลในคอลัมน์อื่นๆ 34](#_Toc179395427)

[3.9.6 การเปรียบเทียบและกรองข้อมูล 34](#_Toc179395428)

[3.9.7 การเรียงลำดับข้อมูล 35](#_Toc179395429)

[3.9.8 การแสดงผล 35](#_Toc179395430)

[3.9.9 ผลลัพธ์ที่ได้ 36](#_Toc179395431)

[3.9.10 ตัวอย่างการเรียกใช้ 36](#_Toc179395432)

[บทที่ 4 อธิบายการทำงานของ code ส่วนการทำงาน 37](#_Toc179395433)

[4.1 การรับคำสั่งจากผู้ใช้ (User Input Handling) 37](#_Toc179395434)

[4.2 โครงสร้างการทำงานของแต่ละเมนู (Menu Functionality) 37](#_Toc179395435)

[4.3 การจัดการข้อผิดพลาด (Error Handling) 38](#_Toc179395436)

[4.4 โครงสร้างข้อมูลและการจัดเก็บ (Data Structure and Storage) 38](#_Toc179395437)

[4.5 การใช้โมดูลเสริม (External Libraries) 38](#_Toc179395438)

[4.6 สรุป (Conclusion) 38](#_Toc179395439)

สารบัญภาพ

ภาพที่ หน้า

[2‑1 หน้าแสดงผลฟังก์ชัน Main 3](#_Toc179395900)

[2‑2 การแสดงผลข้อมูลแบบตัวเลือก 4](#_Toc179395901)

[2‑3 การแสดงผลข้อมูลทั้งหมดแบบค่าเฉลี่ย 4](#_Toc179395902)

[2‑4 การแสดงผลข้อมูลทั้งหมดแบบแสดงทั้ง 4 ไตรมาส 4](#_Toc179395903)

[2‑5 การ Insert data ในกรณีที่ ID พนักงานซ้ำ 5](#_Toc179395904)

[2‑6 การ Insert data ในกรณีที่สามารถ Insert ได้สำเร็จ 5](#_Toc179395905)

[2‑7 หน้า Menu การแก้ไขข้อมูลของพนักงาน 5](#_Toc179395906)

[2‑8 ตัวอย่างการแก้ไขชื่อของพนักงาน 6](#_Toc179395907)

[2‑9 ตัวอย่างการแก้ไขแผนกของพนักงาน 6](#_Toc179395908)

[2‑10 การแก้ไขคะแนนพนักงานโดยการเพิ่มเข้าไปใหม่ 7](#_Toc179395909)

[2‑11 ตัวอย่างการแก้ไขคะแนนพนักงานโดยการเพิ่ม 7](#_Toc179395910)

[2‑12 การแก้ไขคะแนนพนักงานโดยการแทนที่คะแนนที่มีอยู่ 7](#_Toc179395911)

[2‑13 ตัวอย่างการแก้ไขคะแนนพนักงานโดยการแทนที่ 7](#_Toc179395912)

[2‑14 หน้าเมนูการแก้ไขเงินเดือน 8](#_Toc179395913)

[2‑15 ตัวอย่างการเพิ่มเงินเดือน (ถ้าใส่ไป 2000 ก็จะบวกเพิ่มเงินเดือนไป 2000) 8](#_Toc179395914)

[2‑16 ตัวอย่างการลดเงินเดือน (ถ้าใส่ไป 2000 ก็จะลดเงินเดือนไป 2000) 8](#_Toc179395915)

[2‑17 ตัวอย่างการแทนที่เงินเดือน (ถ้าใส่ไป 25000 ก็จะแทนที่เงินเดือนเดิม) 9](#_Toc179395916)

[2‑18 ตัวอย่างการลบข้อมูลพนักงาน 9](#_Toc179395917)

[2‑19 การ Export ข้อมูล 9](#_Toc179395918)

[2‑20 ตัวอย่างข้อมูลที่ทำการ Export 10](#_Toc179395919)

[2‑21 ตัวอย่างการออกจากโปรแกรม 10](#_Toc179395920)

[3‑1 การเตรียมความพร้อมก่อนการเขียนโปรแกรม 11](#_Toc179395921)

[3‑2 การอ่านค่าจาก binary file 12](#_Toc179395922)

[3‑3 การอ่านข้อมูล 12](#_Toc179395923)

[3‑4 การประมวลผลข้อมูลที่อ่านได้ 13](#_Toc179395924)

[3‑5 การจัดการข้อผิดพลาด 13](#_Toc179395925)

[3‑6 การเขียนลงในไฟล์ไบนารี 14](#_Toc179395926)

สารบัญภาพ (ต่อ)

ภาพที่ หน้า

[3‑7 การตรวจสอบความยาวของลิสต์ข้อมูล 14](#_Toc179395927)

[3‑8 หากเข้าเงื่อนไขของเรคคอร์ดแรกในการเขียนข้อมูลลงไฟล์ 14](#_Toc179395928)

[3‑9 ถ้าเรคคอร์ดอื่น ๆ ที่ไม่ใช่เรคคอร์ดแรกในการเขียนข้อมูลลงไฟล์ 15](#_Toc179395929)

[3‑10 ฟังก์ชันของการแก้ไขข้อมูลในไฟล์ไบนารี 16](#_Toc179395930)

[3‑11 ใช้ฟังก์ชัน match เพื่อตรวจสอบว่าผู้ใช้ต้องการแก้ไขคอลัมน์ใด 16](#_Toc179395931)

[3‑12 การแก้ไขคอลัมน์ 'Name' 16](#_Toc179395932)

[3‑13 การวนลูปเพื่อแก้ไขในคอลัมน์ 'Name' 16](#_Toc179395933)

[3‑14 แก้ไขเมื่อพบเรคคอร์ดที่ตรงกับไอดีผู้ใช้ 17](#_Toc179395934)

[3‑15 การตรวจสอบหากข้อมูลใหม่มีจำนวนไม่สอดคล้องกับจำนวนไอดี 17](#_Toc179395935)

[3‑16 การแก้ไขคอลัมน์ 'Department' 17](#_Toc179395936)

[3‑17 การแก้ไขแผนกจะทำงานคล้ายกับการแก้ไขชื่อ 17](#_Toc179395937)

[3‑18 หากข้อมูลแผนกไม่เพียงพอจะใช้ข้อมูลแผนกสุดท้ายที่เหลืออยู่ 17](#_Toc179395938)

[3‑19 การแก้ไขคอลัมน์ 'Score' 18](#_Toc179395939)

[3‑20 เพิ่มคะแนนใหม่เข้าไปในลิสต์ของคะแนนเดิม 18](#_Toc179395940)

[3‑21 แก้ไขคะแนนตามคอลัมน์ที่เลือก 18](#_Toc179395941)

[3‑22 การแก้ไขคอลัมน์ 'Salary' ด้วยตัวเลือก 18](#_Toc179395942)

[3‑23 การเพิ่มเงินเดือน 19](#_Toc179395943)

[3‑24 การลดเงินเดือน 19](#_Toc179395944)

[3‑25 การตั้งค่าเงินเดือนใหม่ 19](#_Toc179395945)

[3‑26 การเพิ่มข้อมูลใหม่ลงในไฟล์ไบนารี 20](#_Toc179395946)

[3‑27 การตรวจสอบ ID ซ้ำ 21](#_Toc179395947)

[3‑28 การเพิ่มข้อมูลใหม่เข้าไปในลิสต์ list\_records 21](#_Toc179395948)

[3‑29 การจะแจ้งข้อความว่า ID ที่ได้รับมาได้ถูกเพิ่มสำเร็จแล้ว 21](#_Toc179395949)

[3‑30 การเปิดไฟล์ในโหมดเพิ่มข้อมูลเพื่อเพิ่มข้อมูลใหม่ลงไฟล์ 21](#_Toc179395950)

[3‑31 การใช้ struct.pack ในการบรรจุข้อมูลใหม่ในรูปแบบไบนารี 22](#_Toc179395951)

[3‑32 การลบข้อมูลในไฟล์ไบนารี 22](#_Toc179395952)

[3‑33 สร้างลิสต์ all\_user\_id ซึ่งประกอบด้วย ID ของผู้ใช้ทั้งหมด 23](#_Toc179395953)

สารบัญภาพ (ต่อ)

ภาพที่ หน้า

[3‑34 การวนลูปตรวจสอบ ID ที่ได้รับจาก multi\_id ว่ามีอยู่ใน all\_user\_id หรือไม่ 23](#_Toc179395954)

[3‑35 การวนลูปเพื่อค้นหาตำแหน่งของแต่ละเรคคอร์ด (index) ที่มี ID ตรงกัน 23](#_Toc179395955)

[3‑36 จัดเรียง list\_index ในลำดับจากมากไปน้อย 23](#_Toc179395956)

[3‑37 ลบเรคคอร์ดจาก list\_records ตามตำแหน่งที่บันทึกใน list\_index 24](#_Toc179395957)

[3‑38 การแสดงผล ID ที่ถูกลบ 24](#_Toc179395958)

[3‑39 การสร้างรายงานสรุปข้อมูลและบันทึกเป็นไฟล์ข้อความ 25](#_Toc179395959)

[3‑40 การอ่านข้อมูลจากไฟล์โดยใช้ numpy 25](#_Toc179395960)

[3‑41 วนลูปในแต่ละเรคคอร์ดใน data\_bin\_file เพื่อดึงข้อมูล 25](#_Toc179395961)

[3‑42 การสร้างแผนกของพนักงานไม่อยู่ในดิกชันนารี departments 26](#_Toc179395962)

[3‑43 การเพิ่มชื่อและคะแนนเฉลี่ยของพนักงานในรายการของแผนก 26](#_Toc179395963)

[3‑44 การรวมคะแนนเฉลี่ยของพนักงานเข้าและเพิ่มจำนวนพนักงานในแผนก 26](#_Toc179395964)

[3‑45 การบันทึกพนักงานที่ทำคะแนนสูงสุดในแผนก 26](#_Toc179395965)

[3‑46 การหาพนักงานของแผนกที่ดีที่สุด 26](#_Toc179395966)

[3‑47 คะแนนเฉลี่ยของแผนกใดมากกว่าคะแนนเฉลี่ยสูงสุดเดิม 26](#_Toc179395967)

[3‑48 การสร้างข้อความรายงาน 27](#_Toc179395968)

[3‑49 บันทึกสตริงรายงานลงในไฟล์ข้อความชื่อ report.txt 27](#_Toc179395969)

[3‑50 การแสดงข้อมูลของพนักงานทั้งหมด 28](#_Toc179395970)

[3‑51 การเลือกโหมดการแสดงข้อมูล 29](#_Toc179395971)

[3‑52 การแสดงข้อมูลในกรณี choice = '1' (แสดงคะแนนเฉลี่ย) 29](#_Toc179395972)

[3‑53 การแสดงข้อมูลในกรณี choice = '2' (แสดงคะแนนแบบแจกแจ้ง) 30](#_Toc179395973)

[3‑54 การตรวจสอบและเรียงลำดับข้อมูล 30](#_Toc179395974)

[3‑55 การตรวจสอบข้อมูล 30](#_Toc179395975)

[3‑56 การแสดงข้อมูลเฉพาะของพนักงานตามคอลัมน์ที่กำหนด 32](#_Toc179395976)

[3‑57 การกำหนดคอลัมน์ที่ต้องการค้นหา 33](#_Toc179395977)

[3‑58 การจัดการข้อมูลในคอลัมน์ Score 34](#_Toc179395978)

[3‑59 การจัดการข้อมูลในคอลัมน์อื่นๆ 34](#_Toc179395979)

[3‑60 ค้นหาคอลัมน์ที่เป็นตัวเลข เช่น Average Score หรือ Salary 35](#_Toc179395980)

สารบัญภาพ (ต่อ)

ภาพที่ หน้า

[3‑61 ค้นหาคอลัมน์ที่เป็นสตริง เช่น ID, Name, หรือ Department 35](#_Toc179395981)

[4‑1 โค้ดแสดงเมนูหลัก 38](#_Toc179395982)

[4‑2 โค้ดแสดงเมนูของการแสดงข้อมูลทั้งหมด 38](#_Toc179395983)

[4‑3 โค้ดแสดงเมนูของการแสดงผลแบบกรองข้อมูล 39](#_Toc179395984)

[4‑4 โค้ดแสดงเมนูของการเพิ่มข้อมูล 39](#_Toc179395985)

[4‑5 รับคำสั่งจากผู้ใช้สำหรับการแก้ไขข้อมูลพนักงานตามคอลัมน์ที่เลือกและตาม ID ที่ป้อน 40](#_Toc179395986)

[4‑6 การรับชื่อใหม่ที่ผู้ใช้ป้อนสำหรับพนักงานตาม ID ที่ต้องการแก้ไข 40](#_Toc179395987)

[4‑7 การแก้ไขข้อมูลแผนกของพนักงานตาม ID ที่ผู้ใช้เลือก 40](#_Toc179395988)

[4‑8 การแก้ไขคะแนน (Score) ของผู้ใช้ 40](#_Toc179395989)

[4‑9 การทำงานย่อยที่ 1 ของการแก้ไขคะแนน (score) 41](#_Toc179395990)

[4‑10 การทำงานย่อยที่ 2 ของการแก้ไขคะแนน (score) 41](#_Toc179395991)

[4‑11 การเพิ่ม, ตัด, และแก้ไขเงินเดือนและแสดงผลคำแนะนำเพื่อให้ผู้ใช้ป้อนข้อมูล 42](#_Toc179395992)

[4‑12 การรับค่า ID (หรือหลาย ID) จากผู้ใช้เพื่อลบข้อมูลที่มี ID เหล่านั้น 42](#_Toc179395993)

[4‑13 การทำรายงานและการแสดงข้อความยืนยันออกจากโปรแกรม 42](#_Toc179395994)

[4‑14 การจัดการข้อผิดพลาดที่อาจเกิดขึ้นในโปรแกรม 42](#_Toc179395995)

# บทนํา

## วัตถุประสงค์ของโครงการ

1. เพื่อจัดเก็บและบริหารข้อมูลพนักงานได้อย่างเป็นระบบ
2. เพื่อค้นหาข้อมูลได้อย่างรวดเร็วและแม่นยำ
3. เพื่อสนับสนุนการตัดสินใจในการบริหารทรัพยากรบุคคล
4. เพื่อสร้างรายงานข้อมูลพนักงานในรูปแบบที่เข้าใจง่าย
5. เพื่อลดข้อผิดพลาดในการจัดการข้อมูลด้วยการตรวจสอบความถูกต้อง
6. เพื่อช่วยประหยัดเวลาและลดความยุ่งยากในการดำเนินงาน

## ขอบเขตของโครงการ

1. ระบบจัดการข้อมูลพนักงานจะมีฟังก์ชันพื้นฐาน 8 ฟังก์ชัน ได้แก่ 1. Main Menu   
   2. Show data 3. Insert data 4. Edit data 5. Show specific 6. Delete Data 7. Exit Data
2. ระบบจัดการข้อมูลพนักงาน ประกอบด้วย 5 ฟิลด์ ได้แก่ 1. ID 2. Name 3. Score 4. Department 5. Salary
3. ระบบจัดการข้อมูลพนักงานมีการจัดเก็บข้อมูลพนักงานและคะแนนพนักงานไว้ใน Text file ชื่อ report.txt ซึ่งมี รหัสพนักงาน ชื่อพนักงาน แผนก คะแนนเฉลี่ย และเงินเดือน

## ประโยชน์ที่ได้รับ

1. พัฒนาระบบจัดการพนักงานได้อย่างมีประสิทธิภาพ
2. พัฒนาทักษะการเขียนโปรแกรม
3. พัฒนาทักษะการทำงานเป็นทีม
4. พัฒนาทักษะการแก้ปัญหาและการจัดการข้อมูล

## เครื่องมือที่ใช้งาน

1. ภาษา Python
2. Microsoft office
3. Visual Code Studio
4. Git Hub
5. Git copilot

# การทำงานของระบบจัดการข้อมูลพนักงาน

## ฟิลด์ในการจัดการข้อมูลพนักงาน

การจัดการข้อมูลพนักงานในระบบประกอบด้วย 5 ฟิลด์หลักๆ ซึ่งแต่ละฟิลด์มีรายละเอียดและความสำคัญดังนี้

### ID (รหัสพนักงาน)

ID เป็นรหัสประจำตัวพนักงานที่ใช้ในการระบุรหัสพนักงานแต่ละคนอย่างเฉพาะเจาะจงซึ่งจะเป็นตัวเลขที่ไม่ซ้ำกันในระบบ ซึ่งรูปแบบของฟิลด์ เป็นประเภทข้อมูล String เช่น 0001, 0002, 0003 เป็นต้น ความสำคัญของฟิลด์นี้คือ เปรียบเสมือน Primary key ที่แต่ละหมายเลขไม่สามารถซ้ำกันได้

### Name (ชื่อพนักงาน)

Name คือ ชื่อของพนักงาน ซึ่งรูปแบบของฟิลด์ เป็นประเภทข้อมูล String (ข้อความ) เช่น “Peter”, “Bob”, “Alice” เป็นต้น ความสำคัญของฟิลด์นี้คือใช้ระบุชื่อของพนักงาน สามารถใช้ค้นหาและสามารถซ้ำกันได้

### Department (แผนก)

Department คือ ชื่อแผนกของพนักงานที่ทำงานใช้ในการระบุว่าพนักงานคนนี้อยู่แผนกอะไรเป็นกลุ่มๆ เช่น “QC”, “IT”, “Finance” เป็นต้น ซึ่งรูปแบบของฟิลด์ เป็นประเภทข้อมูล String ความสำคัญของฟิลด์นี้คือ หมวดหมู่ช่วยในการจัดกรองข้อมูลเป็นแผนกๆสามารถหาข้อมูลได้ง่ายขึ้น

### Score (คะแนน)

Score คือ คะแนนที่พนักงานสอบวัดผลทุกๆ 4 ไตรมาส 1 ไตรมาสจะจัดเก็บ 1 ครั้ง ไว้ประเมินพนักงาน ถ้าพนักงานเข้ามาใหม่จะไม่มีการจัดเก็บคะแนนจะปล่อยให้คะแนนเป็น 0 แต่สามารถเพิ่มหรือแก้ไขได้ จัดเก็บข้อมูลเป็น String แล้วค่อย Convert เป็น List แล้วนำข้อมูลใน List Convert เป็น Float และหาค่าเฉลี่ย ของ 4 ไตรมาส เช่น [“90.0”, “80.0”, “70.0”, “99.0”] เป็นต้น ความสำคัญของฟิลด์นี้คือ ช่วยจัดเก็ยคะแนนของพนักงานเพื่อประเมินหรือขึ้นเงินเดือนให้แก่พนักงาน เพื่อเพิ่มประสิทธิภาพในองค์กร

### Salary (เงินเดือน)

Salary คือ เงินเดือนของพนักงาน จัดเก็บข้อมูลเป็น String แล้วค่อย Convert เป็น Float มีไว้จัดกลุ่มว่าพนักงานที่มีเงินเดือนมากกว่าหรือเท่ากับ 15000 หรือพนักงานที่มีเงินเดือนน้อยกว่า 30000 เป็นต้น

## ฟังก์ชันการทำงานของระบบจัดการพนักงาน

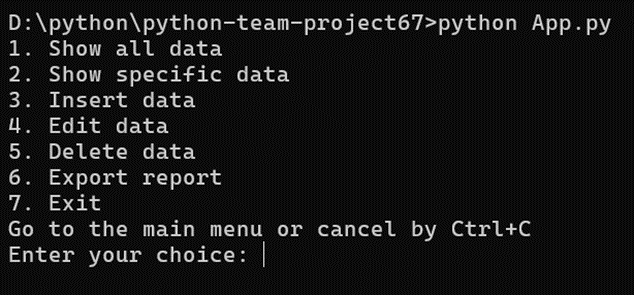
### ฟังก์ชัน main

ฟังก์ชัน main เป็นฟังก์ชันหลักของโปรแกรมระบบจัดการพนักงาน ทำหน้าที่แสดงเมนูการจัดการข้อมูลพนักงาน และรับคำสั่งจากผู้ใช้เพื่อนำไปเรียกใช้ฟังก์ชันย่อยอื่นๆ ตามที่เลือกจากเมนู ฟังก์ชันนี้ทำงานวนซ้ำในลูปจนกว่าผู้ใช้จะเลือกออกจากโปรแกรม โดยรายละเอียดฟงัก์ชันมีดังนี้

* + - 1. โครงสร้างการทำงานของฟังก์ชัน main

แสดงเมนูหลักเมื่อโปรแกรมเริ่มต้น ฟังก์ชัน main() จะทำการแสดงเมนูหลักที่มีตัวเลือกต่างๆ ให้กับผู้ใช้ ผู้ใช้สามารถเลือกตัวเลือกตามหมายเลขที่ต้องการได้ ดังนี้

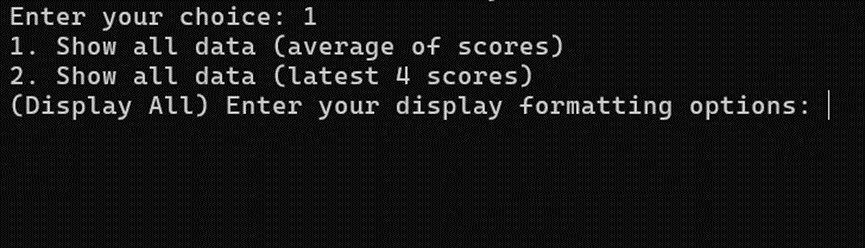
* + - * 1. กด 1 เพื่อแสดงข้อมูลพนักงานทั้งหมด
        2. กด 2 เพื่อค้นหาข้อมูลแบบเฉพาะเจาะจง
        3. กด 3 เพื่อแทรกข้อมูล
        4. กด 4 เพื่อแก้ไขข้อมูล
        5. กด 5 เพื่อลบข้อมูล
        6. กด 6 เพื่อแสดงผลข้อมูลเป็นไฟล์ .txt
        7. กด 7 เพื่อออกจากโปรแกรม



ภาพที่ 2‑1 หน้าแสดงผลฟังก์ชัน Main

### ฟังก์ชัน Show all data

เป็นฟังก์ชันการแสดงผลข้อมูลทั้งหมด โดยสามารถแสดงผลข้อมูลเป็นค่าเฉลี่ยคะแนนของพนักงานหรือจะแสดงผลคะแนนออกมาทั้ง 4 ไตรมาสก็ได้



ภาพที่ 2‑2 การแสดงผลข้อมูลแบบตัวเลือก

A screen shot of a computer

Description automatically generated

ภาพที่ 2‑3 การแสดงผลข้อมูลทั้งหมดแบบค่าเฉลี่ย

A screen shot of a computer

Description automatically generated

ภาพที่ 2‑4 การแสดงผลข้อมูลทั้งหมดแบบแสดงทั้ง 4 ไตรมาส

### ฟังก์ชัน Insert Data

เป็นฟังก์ชันในการแทรกข้อมูลสามารถเลือกแทรกจำนวนข้อมูลพนักงานทั้งหมดกี่คน โดยถ้า ID พนักงานซ้ำจะไม่สามารถแทรกข้อมูลพนักงานได้ โดยข้อมูลที่จะให้แทรก ประกอบด้วย 1.รหัสพนักงาน 2. ชื่อพนักงาน 3. แผนกของพนักงาน 4. คะแนนของพนักงาน (กรณีพนักงานเข้ามาใหม่ไม่ต้องใส่ก็ได้) 5. เงินเดือนของพนักงาน

A black screen with white text

Description automatically generated

ภาพที่ 2‑5 การ Insert data ในกรณีที่ ID พนักงานซ้ำ

A screenshot of a computer screen

Description automatically generated

ภาพที่ 2‑6 การ Insert data ในกรณีที่สามารถ Insert ได้สำเร็จ

### ฟังก์ชัน Edit data

A screen shot of a black background

Description automatically generated

ภาพที่ 2‑7 หน้า Menu การแก้ไขข้อมูลของพนักงาน

เป็นฟังก์ชันที่ใช้สำหรับแก้ไขข้อมูลของพนักงาน โดยจะอ้างอิงจาก ID ซึ่งไม่สามารถแก้ไขได้ โดยสิ่งที่สามารถแก้ไขได้ มีดังนี้

* + - 1. ชื่อของพนักงาน (Name)

การแก้ไขชื่อพนักงาน สามารถแก้ไขได้โดย ใส่ ID พนักงานที่ต้องการจะแก้ไข และแก้ไขชื่อหรือเปลี่ยนชื่อใหม่ ถ้าต้องการแก้ไขข้อมูลพนักงานหลายคนหลังจากใส่เลข ID ของพนักงานคนแรกเสร็จให้เคาะ Space Bar 1 ครั้งและใส่ ID พนักงานคนถัดไป

A screenshot of a computer screen

Description automatically generated

ภาพที่ 2‑8 ตัวอย่างการแก้ไขชื่อของพนักงาน

* + - 1. แผนกของพนักงาน (Department)

การแก้ไขแผนกของพนักงาน สามารถแก้ไขได้โดย ใส่ ID พนักงานที่ต้องการจะแก้ไข และแก้ไขชื่อแผนกหรือเปลี่ยนชื่อแผนกใหม่ ถ้าต้องการแก้ไขข้อมูลพนักงานหลายคนหลังจากใส่เลข ID ของพนักงานคนแรกเสร็จให้เคาะ Space Bar 1 ครั้งและใส่ ID พนักงานคนถัดไป

A screenshot of a computer screen

Description automatically generated

ภาพที่ 2‑9 ตัวอย่างการแก้ไขแผนกของพนักงาน

* + - 1. คะแนนของพนักงาน (Score)

การแก้ไขข้อมูลคะแนนของพนักงาน สามารถแก้ไขได้โดย ใส่ ID พนักงานที่ต้องการจะแก้ไข และแก้ไขคะแนน การแก้ไขคะแนนมี 2 แบบคือ แบบที่ 1 คือ การเพิ่มคะแนนเข้าไปใหม่โดยจะเพิ่มไปที่ Index หลังสุดและคะแนนข้างหน้าก็จะหายไป แบบที่ 2 คือ แก้ไขคะแนนที่มีอยู่โดยระบุ Index ที่ต้องการจะแก้ไขและใส่คะแนนเข้าไปใหม่ ถ้าต้องการแก้ไขข้อมูลพนักงานหลายคนหลังจากใส่เลข ID ของพนักงานคนแรกเสร็จให้เคาะ Space Bar 1 ครั้งและใส่ ID พนักงานคนถัดไป

A screenshot of a computer program

Description automatically generated

ภาพที่ 2‑10 การแก้ไขคะแนนพนักงานโดยการเพิ่มเข้าไปใหม่

A black background with white text

Description automatically generated

ภาพที่ 2‑11 ตัวอย่างการแก้ไขคะแนนพนักงานโดยการเพิ่ม

A screenshot of a computer program

Description automatically generated

ภาพที่ 2‑12 การแก้ไขคะแนนพนักงานโดยการแทนที่คะแนนที่มีอยู่

A black background with white letters

Description automatically generated

ภาพที่ 2‑13 ตัวอย่างการแก้ไขคะแนนพนักงานโดยการแทนที่

* + - 1. เงินเดือนของพนักงาน (Salary)

การแก้ไขข้อมูลเงินเดือนของพนักงาน สามารถแก้ไขได้โดย ใส่ ID พนักงานที่ต้องการจะแก้ไข และแก้ไขข้อมูลเงินเดือน การแก้ไขเงินเดือนประกอบด้วย 3 ฟังก์ชัน คือ   
1. การเพิ่มเงินเดือน 2.การลดเงินเดือน 3.การแทนที่เงินเดือนที่มีอยู่ ถ้าต้องการแก้ไขข้อมูลพนักงานหลายคนหลังจากใส่เลข ID ของพนักงานคนแรกเสร็จให้เคาะ Space Bar 1 ครั้งและใส่ ID พนักงานคนถัดไป

A screenshot of a computer program

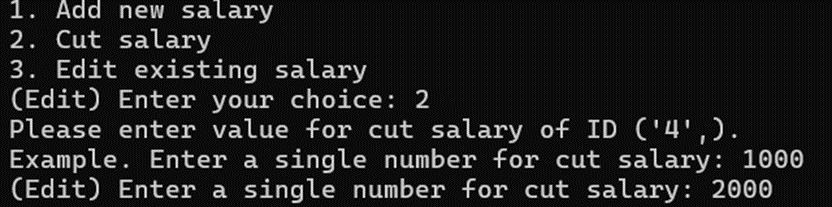
Description automatically generated

ภาพที่ 2‑14 หน้าเมนูการแก้ไขเงินเดือน

A black screen with white text

Description automatically generated

ภาพที่ 2‑15 ตัวอย่างการเพิ่มเงินเดือน (ถ้าใส่ไป 2000 ก็จะบวกเพิ่มเงินเดือนไป 2000)



ภาพที่ 2‑16 ตัวอย่างการลดเงินเดือน (ถ้าใส่ไป 2000 ก็จะลดเงินเดือนไป 2000)

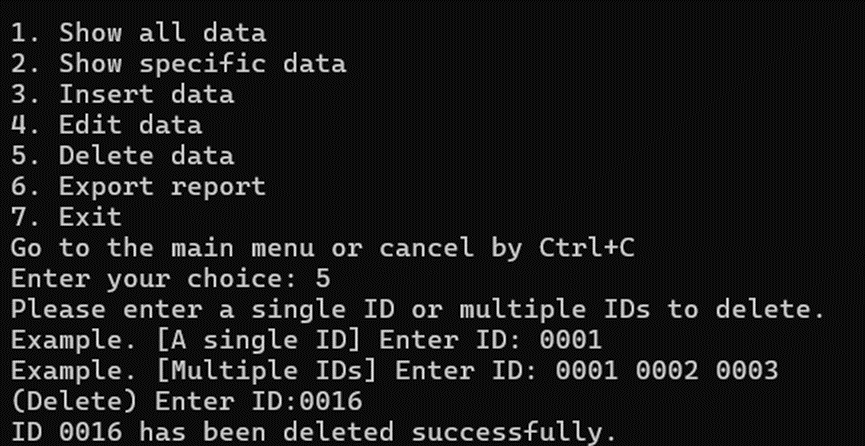
A black screen with white text

Description automatically generated

ภาพที่ 2‑17 ตัวอย่างการแทนที่เงินเดือน (ถ้าใส่ไป 25000 ก็จะแทนที่เงินเดือนเดิม)

### ฟังก์ชัน Delete

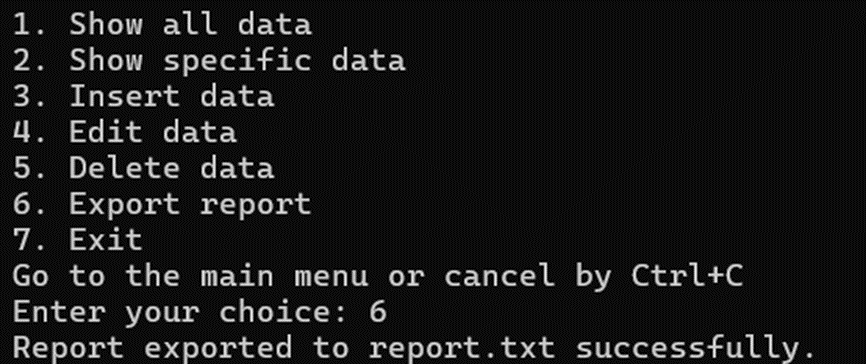
เป็นฟังก์ชันในการลบข้อมูลจะทำการลบข้อมูลของพนักงานโดยระบุ ID ของพนักงานและทำการลบข้อมูลภายในฟิลด์พนักงานคนนั้นทั้งหมด



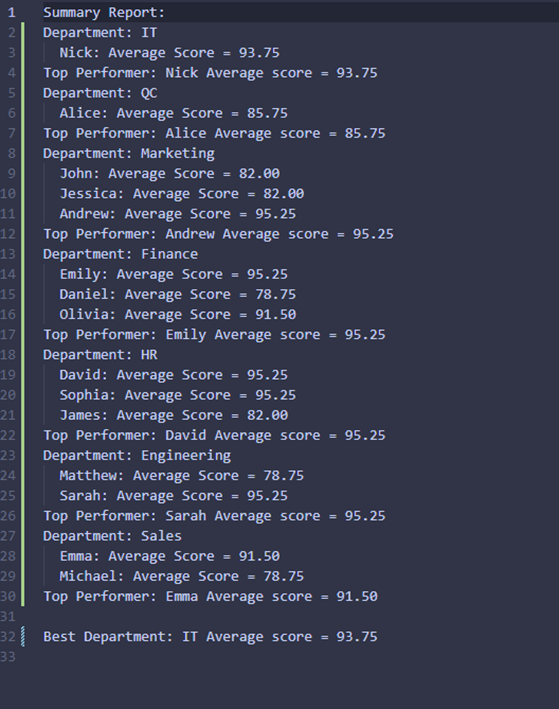
ภาพที่ 2‑18 ตัวอย่างการลบข้อมูลพนักงาน

### ฟังก์ชัน Export

เป็นฟังก์ชันในการ Export ข้อมูลออกมาเป็นไฟล์ชื่อว่า Report.txt



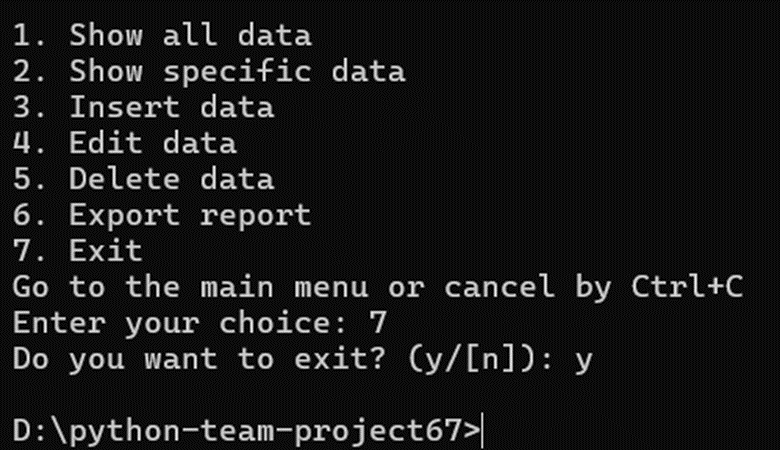
ภาพที่ 2‑19 การ Export ข้อมูล



ภาพที่ 2‑20 ตัวอย่างข้อมูลที่ทำการ Export

### ฟังก์ชัน Exit

เป็นฟังก์ชันสำหรับการออกจากโปรแกรม โดยเมื่อเลือกฟังก์ชันแล้วตัวโปรแกรมจะทำการถามว่าต้องการออกจากโปรแกรมหรือไม่ถ้าต้องการออกให้เลือก Y เพื่อทำการออกจากโปรแกรม



ภาพที่ 2‑21 ตัวอย่างการออกจากโปรแกรม

# อธิบายการทำงานของ code ส่วนการประมวลผล

## การเตรียมความพร้อมก่อนการเขียนโปรแกรม

A black screen with text

Description automatically generated

ภาพที่ 3‑1 การเตรียมความพร้อมก่อนการเขียนโปรแกรม

### การนำเข้าโมดูลและการกำหนดค่าเริ่มต้น

* + - 1. โปรแกรมได้ทำการนำเข้าโมดูล struct, numpy และ pandas เพื่อใช้ในการจัดการข้อมูลและการประมวลผลต่างๆ ในโปรแกรม
      2. โมดูล struct: ใช้สำหรับการอ่านและเขียนข้อมูลในรูปแบบไบนารี (Binary Format)
      3. โมดูล numpy: ใช้สำหรับการจัดการข้อมูลในลักษณะอาเรย์ (Array) และการประมวลผลข้อมูลเชิงตัวเลข
      4. โมดูล pandas: ใช้สำหรับการจัดการข้อมูลในรูปแบบ DataFrame ซึ่งช่วยให้สามารถแสดงผลและจัดการข้อมูลได้สะดวกขึ้น

### การกำหนดเส้นทางของไฟล์ไบนารี

bin\_path = "data.bin" เป็นการกำหนดชื่อไฟล์และเส้นทางสำหรับการจัดเก็บข้อมูลในรูปแบบไบนารี ซึ่งข้อมูลพนักงานทั้งหมดจะถูกจัดเก็บและเข้าถึงผ่านไฟล์ data.bin นี้

## การอ่านค่าจากไฟล์ไบนารี

ใช้ฟังก์ชันที่ชื่อว่า readDataFromBinFile ทำหน้าที่อ่านข้อมูลจากไฟล์ไบนารีและแปลงข้อมูลให้อยู่ในรูปแบบของลิสต์ (list) ซึ่งฟังก์ชันจะคืนค่าลิสต์ของเรคคอร์ดทั้งหมดที่ถูกอ่านจากไฟล์

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑2 การอ่านค่าจาก binary file

### การประกาศตัวแปรไว้เก็บข้อมูลไปประมวลผล

**list\_records:** เป็นลิสต์ที่จะเก็บเรคคอร์ดที่ถูกอ่านมาจากไฟล์

### การเปิดไฟล์

ฟังก์ชันนี้ใช้ with open(bin\_path, "rb") เพื่อเปิดไฟล์ไบนารีในโหมดการอ่าน (rb ย่อมาจาก read binary) ซึ่งจะป้องกันไม่ให้เกิดการลืมปิดไฟล์

### การอ่านข้อมูลก่อนนำไปประมวลผล

* + - 1. record\_size = struct.calcsize("20s20s20s20sf") คำนวณขนาดของเรคคอร์ดแต่ละเรคคอร์ดในไฟล์ โดยมีรูปแบบข้อมูลเป็น string ขนาด 20 bytes ทั้งหมด 4 ตัว และตัวเลขแบบ float 1 ตัว (20s20s20s20sf)
      2. วนลูปอ่านข้อมูลด้วย file.read(record\_size) เพื่ออ่านข้อมูลทีละเรคคอร์ดตามขนาดที่คำนวณได้
      3. หากอ่านได้ข้อมูลไม่ครบ (ไม่มีข้อมูลเหลืออยู่ในไฟล์) การอ่านจะหยุดทันทีด้วย if not data: break

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑3 การอ่านข้อมูล

### การประมวลผลข้อมูลที่อ่านได้

* + - 1. struct.unpack("20s20s20s20sf", data) ทำการแปลงข้อมูลที่อ่านจากไฟล์ให้เป็นรูปแบบ tuple โดยแต่ละสมาชิกของ tuple จะสอดคล้องกับรูปแบบข้อมูลที่กำหนดไว้ในฟอร์แมต
      2. record\_3 ทำการแปลงข้อมูลใน field ที่ 3 (ตัวที่สี่ใน tuple) จาก string ไปเป็นลิสต์ของตัวเลข float (ในกรณีที่ข้อมูลไม่ครบ 4 ตัว จะเติมตัวเลข 0.0 ให้ครบ 4 ตัว)
      3. นำข้อมูลที่แปลงได้มาจัดรูปแบบใหม่เป็นลิสต์โดยทำการตัดสัญลักษณ์ '\x00' ที่ติดมากับ string ในข้อมูลแต่ละส่วน และเก็บไว้ในลิสต์ list\_records

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑4 การประมวลผลข้อมูลที่อ่านได้

### การจัดการข้อผิดพลาด

หากไฟล์ที่ต้องการอ่านไม่พบ (FileNotFoundError) ฟังก์ชันจะสร้างไฟล์ใหม่ขึ้นมาและทำการเขียนไฟล์เปล่าๆ ลงไป ซึ่งจำเป็นต้องใช้คำสั่ง try ถึงจะสามารถใช้คำสั่ง except ได้

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑5 การจัดการข้อผิดพลาด

### ผลลัพธ์ที่ได้จากการเรียกใช้ฟังก์ชัน

ฟังก์ชันจะคืนค่าลิสต์ list\_records ที่เก็บเรคคอร์ดทั้งหมดในรูปแบบที่แปลงเสร็จแล้ว

### ตัวอย่างการเรียกใช้

การเรียกใช้ฟังก์ชันนี้คือ readDataFromBinFile() ซึ่งจะคืนค่าลิสต์ของข้อมูลที่อ่านจากไฟล์

## การเขียนลงในไฟล์ไบนารี

ฟังก์ชันนี้ชื่อว่า writeDataToBinFile มีหน้าที่เขียนข้อมูลที่ได้รับมาในรูปแบบของลิสต์ (list) ลงในไฟล์ไบนารี โดยทำการบันทึกข้อมูลแต่ละเรคคอร์ดที่มีรูปแบบคงที่

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑6 การเขียนลงในไฟล์ไบนารี

### การตรวจสอบความยาวของลิสต์ข้อมูล

ฟังก์ชันจะรับข้อมูล list\_records ที่เป็นลิสต์ของเรคคอร์ด ถ้าลิสต์นี้ไม่มีข้อมูล (ความยาวของลิสต์เป็น 0) ฟังก์ชันจะเปิดไฟล์ในโหมดเขียนไบนารี (wb) แล้วทำการเขียนข้อมูลเปล่า (file.write(b'')) ลงไปเพื่อเคลียร์ข้อมูลเดิมในไฟล์ จากนั้นจะจบการทำงาน

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑7 การตรวจสอบความยาวของลิสต์ข้อมูล

### การวนลูปเขียนข้อมูล

ฟังก์ชันจะวนลูปผ่านแต่ละเรคคอร์ดใน list\_records โดยใช้ฟังก์ชัน enumerate() เพื่อให้สามารถเข้าถึงทั้งดัชนี (index) และเรคคอร์ด (record) ได้ ดังนี้:

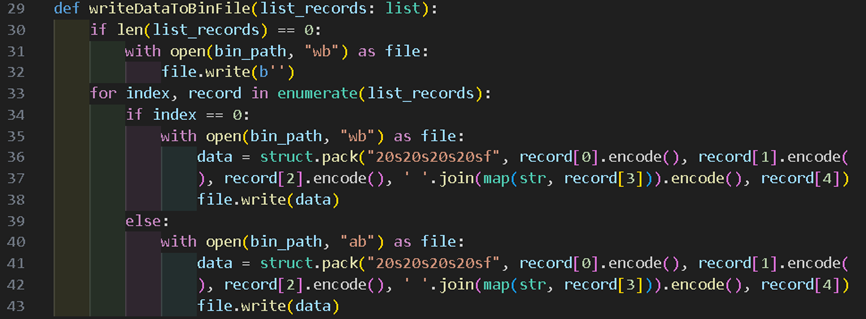
* + - 1. ถ้า index == 0 (เรคคอร์ดแรก):
         1. ฟังก์ชันจะเปิดไฟล์ในโหมดเขียนไบนารี (wb) เพื่อเขียนทับข้อมูลเดิมทั้งหมด
         2. จากนั้นจะทำการบรรจุข้อมูล (pack) เรคคอร์ดในรูปแบบไบนารีด้วย struct.pack("20s20s20s20sf", ...) ซึ่งแต่ละสมาชิกของเรคคอร์ดจะถูกแปลงเป็นข้อมูลชนิดไบนารีที่กำหนดไว้ (string ขนาด 20 bytes ทั้งหมด 4 ตัว และ float 1 ตัว)
         3. สุดท้ายจะทำการเขียนข้อมูลที่ได้ลงไปในไฟล์ด้วย file.write(data)

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑8 หากเข้าเงื่อนไขของเรคคอร์ดแรกในการเขียนข้อมูลลงไฟล์

* + - 1. ถ้า index > 0 (เรคคอร์ดอื่น ๆ ที่ไม่ใช่เรคคอร์ดแรก):
         1. ฟังก์ชันจะเปิดไฟล์ในโหมดเพิ่มข้อมูล (ab) ซึ่งจะทำการเขียนข้อมูลต่อท้ายจากข้อมูลที่มีอยู่ในไฟล์
         2. กระบวนการแปลงข้อมูล (pack) และการเขียนข้อมูลลงในไฟล์จะเหมือนกับกรณีของเรคคอร์ดแรก



ภาพที่ 3‑9 ถ้าเรคคอร์ดอื่น ๆ ที่ไม่ใช่เรคคอร์ดแรกในการเขียนข้อมูลลงไฟล์

### การจัดรูปแบบข้อมูลก่อนเขียน

ข้อมูลแต่ละเรคคอร์ดจะถูกแปลงให้เป็นข้อมูลไบนารีผ่านฟังก์ชัน struct.pack โดย:

* + - 1. ข้อมูลตัวอักษร (strings) จะถูกเข้ารหัส (encode) เป็นข้อมูลไบนารี
      2. ข้อมูลที่เป็นลิสต์ของตัวเลข (record[3]) จะถูกแปลงเป็น string โดยใช้ join(map(str, record[3])) แล้วเข้ารหัสเป็นไบนารี
      3. ข้อมูล float (record[4]) จะถูกเก็บโดยตรงในรูปแบบ float ในไบนารี

### ผลลัพธ์ที่ได้

ฟังก์ชันจะทำการเขียนข้อมูลที่ได้รับมาในรูปแบบไบนารีลงในไฟล์ตามรูปแบบที่กำหนด โดยการเขียนข้อมูลเรคคอร์ดจะเขียนทีละเรคคอร์ดต่อเนื่องกันในไฟล์

### ตัวอย่างการเรียกใช้

ฟังก์ชันนี้สามารถเรียกใช้ด้วย writeDataToBinFile(list\_records) โดยที่ list\_records เป็นลิสต์ของเรคคอร์ดที่ต้องการเขียนลงในไฟล์

## การแก้ไขข้อมูลในไฟล์ไบนารี

ฟังก์ชันนี้ชื่อว่า editData มีหน้าที่แก้ไขข้อมูลในไฟล์ไบนารีตามคอลัมน์ที่ต้องการแก้ไข โดยรับข้อมูลระบุว่าจะแก้ไขคอลัมน์ไหน (pointed\_col), ข้อมูลระบุผู้ใช้ที่จะแก้ไข (id), ข้อมูลใหม่ที่จะแก้ไข (new\_data), ตัวเลือกการแก้ไข (choice\_edit), และคอลัมน์ที่เลือก (selected\_col)

A screen shot of a computer

Description automatically generated

ภาพที่ 3‑10 ฟังก์ชันของการแก้ไขข้อมูลในไฟล์ไบนารี

### การอ่านข้อมูลก่อนแก้ไขข้อมูล

ฟังก์ชันเริ่มต้นด้วยการอ่านข้อมูลทั้งหมดจากไฟล์ไบนารีด้วยการเรียกใช้ฟังก์ชัน readDataFromBinFile() และเก็บข้อมูลไว้ในตัวแปร list\_records ซึ่งเป็นลิสต์ของเรคคอร์ด

### การเลือกคอลัมน์ที่จะแก้ไขข้อมูล



ภาพที่ 3‑11 ใช้ฟังก์ชัน match เพื่อตรวจสอบว่าผู้ใช้ต้องการแก้ไขคอลัมน์ใด

ใช้ฟังก์ชัน match เพื่อตรวจสอบว่าผู้ใช้ต้องการแก้ไขคอลัมน์ใด โดย pointed\_col จะระบุคอลัมน์ที่ต้องการแก้ไข ซึ่งสามารถเลือกได้ระหว่าง:

* + - 1. 'Name' (ชื่อ)
      2. 'Department' (แผนก)
      3. 'Score' (คะแนน)
      4. 'Salary' (เงินเดือน)

### กรณีการแก้ไขตามคอลัมน์

A black screen with text

Description automatically generated

ภาพที่ 3‑12 การแก้ไขคอลัมน์ 'Name'

* + - 1. การแก้ไขคอลัมน์ 'Name'
         1. ฟังก์ชันจะวนลูปตามจำนวนผู้ใช้ (id) และค้นหาเรคคอร์ดที่มีรหัส (record[0]) ตรงกับ id ที่ได้รับมา

A black screen with text

Description automatically generated

ภาพที่ 3‑13 การวนลูปเพื่อแก้ไขในคอลัมน์ 'Name'

* + - * 1. ถ้าพบ เรคคอร์ดนั้นจะถูกแก้ไขชื่อ (คอลัมน์ 1 หรือ record[1]) ด้วยข้อมูลใหม่ (new\_data[i])

A black screen with text

Description automatically generated

ภาพที่ 3‑14 แก้ไขเมื่อพบเรคคอร์ดที่ตรงกับไอดีผู้ใช้

* + - * 1. มีการตรวจสอบหากข้อมูลใหม่ไม่เพียงพอ จะส่งข้อผิดพลาดกลับมา

A black screen with text

Description automatically generated

ภาพที่ 3‑15 การตรวจสอบหากข้อมูลใหม่มีจำนวนไม่สอดคล้องกับจำนวนไอดี

* + - 1. การแก้ไขคอลัมน์ 'Department'

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑16 การแก้ไขคอลัมน์ 'Department'

* + - * 1. ฟังก์ชันจะทำงานคล้ายกับการแก้ไขชื่อ โดยแก้ไขข้อมูลในคอลัมน์ 2 หรือ record[2] (แผนก)

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑17 การแก้ไขแผนกจะทำงานคล้ายกับการแก้ไขชื่อ

* + - * 1. ถ้าข้อมูลไม่เพียงพอ ฟังก์ชันจะใช้ข้อมูลสุดท้ายที่เหลืออยู่

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑18 หากข้อมูลแผนกไม่เพียงพอจะใช้ข้อมูลแผนกสุดท้ายที่เหลืออยู่

* + - 1. การแก้ไขคอลัมน์ 'Score'

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑19 การแก้ไขคอลัมน์ 'Score'

ฟังก์ชันตรวจสอบการแก้ไขด้วยตัวเลือก choice\_edit ดังนี้:

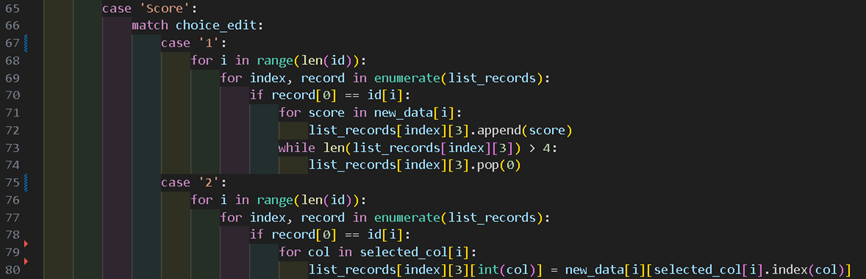
* + - * 1. '1': เพิ่มคะแนนใหม่เข้าไปในลิสต์ของคะแนน (record[3]) โดยเพิ่มไปเรื่อย ๆ จนเกิน 4 คะแนน จากนั้นจะลบคะแนนแรกๆ ออกเพื่อให้เหลือ 4 คะแนน

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑20 เพิ่มคะแนนใหม่เข้าไปในลิสต์ของคะแนนเดิม

* + - * 1. '2': แก้ไขคะแนนตามคอลัมน์ที่เลือก (selected\_col) โดยใช้ข้อมูลใหม่จาก new\_data[i] และอัปเดตคะแนนในคอลัมน์ที่เลือก



ภาพที่ 3‑21 แก้ไขคะแนนตามคอลัมน์ที่เลือก

* + - 1. การแก้ไขคอลัมน์ 'Salary'

A screen shot of a computer

Description automatically generated

ภาพที่ 3‑22 การแก้ไขคอลัมน์ 'Salary' ด้วยตัวเลือก

ฟังก์ชันตรวจสอบการแก้ไขด้วยตัวเลือก choice\_edit ดังนี้:

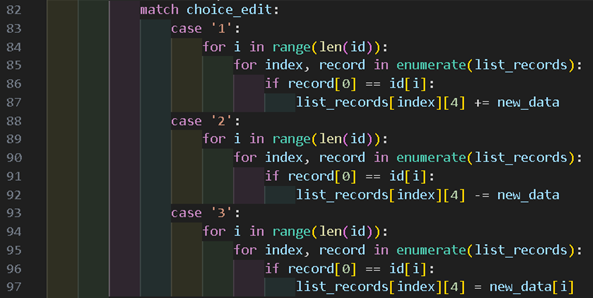
* + - * 1. '1': เพิ่มเงินเดือน

A screen shot of a computer

Description automatically generated

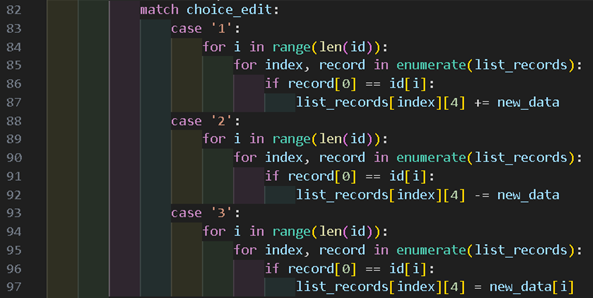
ภาพที่ 3‑23 การเพิ่มเงินเดือน

* + - * 1. '2': ลดเงินเดือน



ภาพที่ 3‑24 การลดเงินเดือน

* + - * 1. '3': ตั้งค่าเงินเดือนใหม่



ภาพที่ 3‑25 การตั้งค่าเงินเดือนใหม่

### การจัดการข้อมูลที่ไม่ถูกต้อง

หาก pointed\_col ไม่ตรงกับตัวเลือกที่รองรับ ฟังก์ชันจะพิมพ์ข้อความแจ้งเตือนว่าทางเลือกไม่ถูกต้อง และจะสิ้นสุดการทำงาน

### การเขียนข้อมูลกลับลงไฟล์

เมื่อแก้ไขข้อมูลเสร็จแล้ว ฟังก์ชันจะเขียนข้อมูลที่ถูกแก้ไขกลับลงไปในไฟล์ไบนารีด้วยการเรียกใช้ฟังก์ชัน writeDataToBinFile(list\_records)

### ผลลัพธ์ที่ได้

ฟังก์ชันนี้จะทำการแก้ไขข้อมูลในไฟล์ตามคอลัมน์ที่ผู้ใช้ระบุ โดยผลลัพธ์คือข้อมูลที่ถูกแก้ไขแล้วจะถูกเขียนกลับลงไปในไฟล์ไบนารี

### ตัวอย่างการเรียกใช้

ฟังก์ชัน editData(pointed\_col: str, id: tuple, new\_data, choice\_edit:str=None, selected\_col:list=None) สามารถเรียกใช้ได้ดังนี้:

* + - 1. การแก้ไขคอลัมน์ 'Name':

editData('Name', ['0001', '0002', '0003'], ['Silfy', 'Vyne', 'Buck'])

* + - 1. การแก้ไขคอลัมน์ 'Department'

editData('Department', ['0001', '0002', '0003'], ['IT', 'HR', 'IT'])

* + - 1. การแก้ไขคอลัมน์ 'Score'

editData('Score', ['0001', '0002', '0003'], [[97, 98, 99], [45, 46, 47], [50, 60, 70]], '2', [['1', '3', '2'], ['1', '2', '3'], ['3', '2', '1']])

* + - 1. การแก้ไขคอลัมน์ 'Salary'

editData('Salary', ['0001', '0002', '0003'], [10000, 20000, 30000], '3')

## การเพิ่มข้อมูลใหม่ลงในไฟล์ไบนารี

ฟังก์ชันนี้ชื่อว่า addData มีหน้าที่เพิ่มเรคคอร์ดใหม่เข้าไปในไฟล์ไบนารี โดยรับข้อมูลใหม่ 5 ชุด ได้แก่ id (รหัส), name (ชื่อ), department (แผนก), score (คะแนนในรูปแบบลิสต์), และ salary (เงินเดือน)

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑26 การเพิ่มข้อมูลใหม่ลงในไฟล์ไบนารี

### การอ่านข้อมูลจากไฟล์เพื่อตรวจสอบ ID ซ้ำซ้อน

ฟังก์ชันเริ่มต้นด้วยการอ่านข้อมูลจากไฟล์ไบนารีโดยเรียกใช้ฟังก์ชัน readDataFromBinFile() และเก็บข้อมูลทั้งหมดไว้ในตัวแปร list\_records

### การตรวจสอบ ID ซ้ำ

ฟังก์ชันจะตรวจสอบว่ามี id ที่ได้รับมาอยู่ใน list\_records แล้วหรือไม่

* + - 1. ใช้การวนลูปเพื่อเช็กว่า record[0] (ซึ่งเก็บ ID ของแต่ละเรคคอร์ด) ตรงกับ id ที่ได้รับมา
      2. ถ้าพบว่ามี ID ซ้ำ ฟังก์ชันจะส่งข้อผิดพลาดโดยการ raise Exception และพิมพ์ข้อความแจ้งว่ามี ID ซ้ำแล้ว ซึ่งจะหยุดการทำงานของฟังก์ชัน

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑27 การตรวจสอบ ID ซ้ำ

### การเพิ่มข้อมูลใหม่

ถ้าไม่มีข้อผิดพลาดเกิดขึ้น (ไม่มี ID ซ้ำ)

* + - 1. ข้อมูลใหม่จะถูกเพิ่มเข้าไปในลิสต์ list\_records ในรูปแบบของลิสต์ที่มี 5 คอลัมน์ ได้แก่ id, name, department, score, และ salary

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑28 การเพิ่มข้อมูลใหม่เข้าไปในลิสต์ list\_records

* + - 1. ฟังก์ชันจะแจ้งข้อความว่า ID ที่ได้รับมาได้ถูกเพิ่มสำเร็จแล้ว (ID {id} has been added successfully.)

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑29 การจะแจ้งข้อความว่า ID ที่ได้รับมาได้ถูกเพิ่มสำเร็จแล้ว

### การเขียนข้อมูลใหม่ลงไฟล์

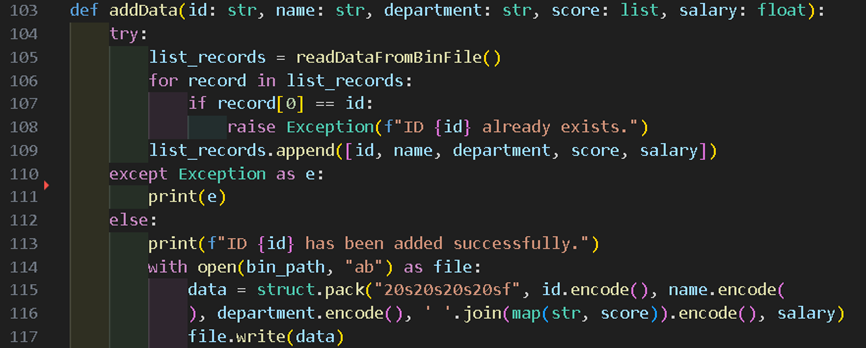
* + - 1. ฟังก์ชันจะเปิดไฟล์ในโหมดเพิ่มข้อมูล (ab) ซึ่งเป็นการเขียนข้อมูลต่อท้ายไฟล์โดยไม่ลบข้อมูลเดิม

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑30 การเปิดไฟล์ในโหมดเพิ่มข้อมูลเพื่อเพิ่มข้อมูลใหม่ลงไฟล์

* + - 1. ใช้ struct.pack("20s20s20s20sf", ...) ในการบรรจุข้อมูลใหม่ในรูปแบบไบนารี:
         1. id, name, และ department จะถูกแปลงเป็น string แล้วเข้ารหัสเป็นไบนารีด้วย .encode()
         2. score ซึ่งเป็นลิสต์ของคะแนนจะถูกแปลงเป็น string ด้วยการใช้ join(map(str, score)) แล้วเข้ารหัสเป็นไบนารี
         3. salary จะถูกบรรจุเป็นตัวเลข float ตามรูปแบบที่กำหนด



ภาพที่ 3‑31 การใช้ struct.pack ในการบรรจุข้อมูลใหม่ในรูปแบบไบนารี

* + - 1. ข้อมูลที่ถูกบรรจุเป็นไบนารีจะถูกเขียนลงไปในไฟล์ด้วย file.write(data)

### ผลลัพธ์ที่ได้

ฟังก์ชันนี้จะเพิ่มข้อมูลใหม่เข้าไปในไฟล์ไบนารี โดยจะทำการตรวจสอบว่า ID นั้นมีอยู่แล้วหรือไม่ ถ้าไม่มี ID ซ้ำ ข้อมูลใหม่จะถูกเพิ่มลงไฟล์

### ตัวอย่างการเรียกใช้

ตัวอย่างนี้จะเพิ่มเรคคอร์ดใหม่สำหรับผู้ใช้ที่มี ID "002" ชื่อ "Jane Smith" ทำงานในแผนก HR มีคะแนน และเงินเดือนตามที่ระบุ: addData("002", "Jane Smith", "HR", [80.5, 90.0, 85.0, 70.0], 50000.0)

## การลบข้อมูลในไฟล์ไบนารี

ฟังก์ชันนี้ชื่อว่า deleteData มีหน้าที่ลบเรคคอร์ดในไฟล์ไบนารีตามรหัส (ID) ที่ได้รับจากลิสต์ multi\_id ซึ่งเป็นลิสต์ของ ID ที่ต้องการลบ

A screen shot of a computer program

Description automatically generated

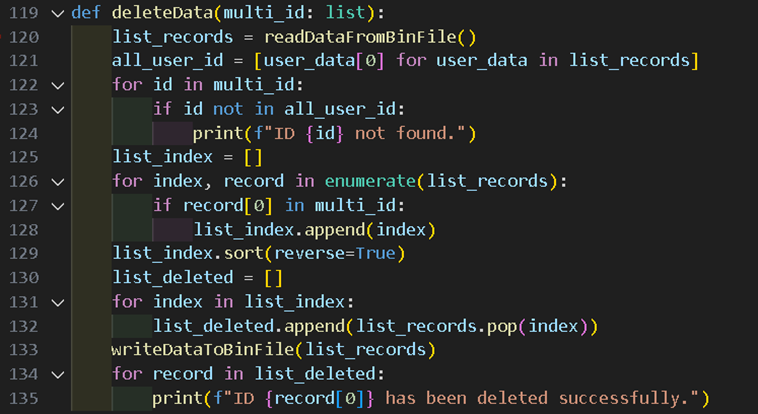
ภาพที่ 3‑32 การลบข้อมูลในไฟล์ไบนารี

### การอ่านข้อมูลก่อนลบข้อมูล

ฟังก์ชันเริ่มต้นด้วยการอ่านข้อมูลทั้งหมดจากไฟล์ไบนารีโดยใช้ฟังก์ชัน readDataFromBinFile() และเก็บข้อมูลทั้งหมดไว้ในตัวแปร list\_records ซึ่งเป็นลิสต์ของเรคคอร์ด

### การตรวจสอบ ID ที่จะลบ

* + - 1. สร้างลิสต์ all\_user\_id ซึ่งประกอบด้วย ID ของผู้ใช้ทั้งหมดใน list\_records โดยใช้การสร้างลิสต์อย่างย่อ (list comprehension) จาก record[0] ซึ่งเป็นคอลัมน์ ID ของแต่ละเรคคอร์ด



ภาพที่ 3‑33 สร้างลิสต์ all\_user\_id ซึ่งประกอบด้วย ID ของผู้ใช้ทั้งหมด

* + - 1. วนลูปตรวจสอบ ID ที่ได้รับจาก multi\_id ว่ามีอยู่ใน all\_user\_id หรือไม่:

หาก ID ไม่พบใน list\_records ฟังก์ชันจะแสดงข้อความว่า ID {id} not found. แต่จะยังคงดำเนินการลบ ID อื่นๆ ที่พบต่อไป

A screen shot of a computer program

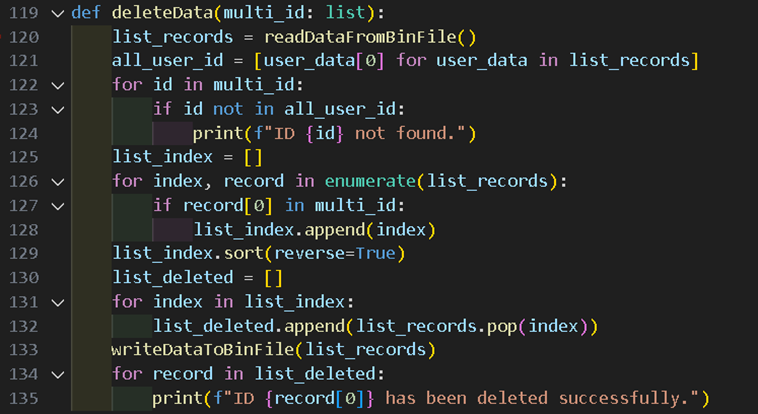
Description automatically generated

ภาพที่ 3‑34 การวนลูปตรวจสอบ ID ที่ได้รับจาก multi\_id ว่ามีอยู่ใน all\_user\_id หรือไม่

### การลบเรคคอร์ด

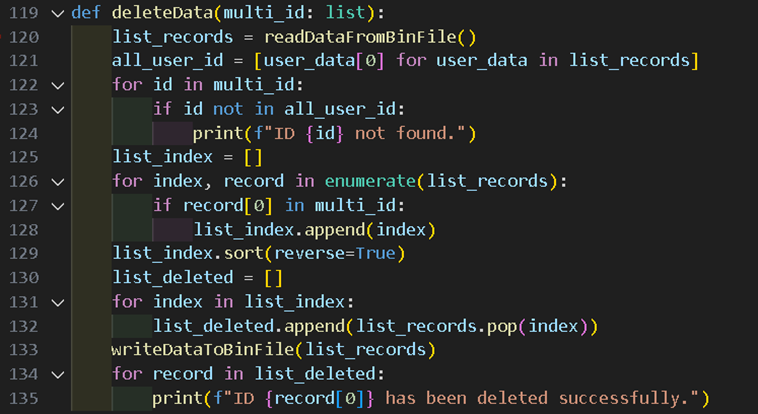
* + - 1. สร้างลิสต์ list\_index ซึ่งเก็บตำแหน่งของเรคคอร์ดที่มี ID ตรงกับ ID ที่ได้รับจาก multi\_id

ใช้การวนลูปเพื่อค้นหาตำแหน่งของแต่ละเรคคอร์ด (index) ที่มี ID ตรงกัน และเก็บค่า index ไว้ใน list\_index



ภาพที่ 3‑35 การวนลูปเพื่อค้นหาตำแหน่งของแต่ละเรคคอร์ด (index) ที่มี ID ตรงกัน

* + - 1. จัดเรียง list\_index ในลำดับจากมากไปน้อย (จาก index สูงไปต่ำ) โดยใช้ sort(reverse=True) เพื่อป้องกันปัญหาในการลบเรคคอร์ดหลายตัวจากลิสต์ ซึ่งอาจทำให้ตำแหน่งของเรคคอร์ดที่เหลือเปลี่ยนไป



ภาพที่ 3‑36 จัดเรียง list\_index ในลำดับจากมากไปน้อย

* + - 1. ลบเรคคอร์ดจาก list\_records ตามตำแหน่งที่บันทึกใน list\_index โดยใช้ pop() และเก็บเรคคอร์ดที่ถูกลบไว้ใน list\_deleted

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑37 ลบเรคคอร์ดจาก list\_records ตามตำแหน่งที่บันทึกใน list\_index

### การเขียนข้อมูลกลับลงไฟล์

ฟังก์ชันเขียนข้อมูลที่เหลืออยู่ใน list\_records หลังจากการลบ กลับลงไปในไฟล์ไบนารี โดยเรียกใช้ฟังก์ชัน writeDataToBinFile(list\_records)

### การแสดงผล ID ที่ถูกลบ

ฟังก์ชันจะแสดงข้อความว่ารหัส ID แต่ละตัวที่ถูกลบสำเร็จ (ID {record[0]} has been deleted successfully.) โดยใช้ข้อมูลจากลิสต์ list\_deleted ซึ่งเก็บเรคคอร์ดที่ถูกลบออก

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑38 การแสดงผล ID ที่ถูกลบ

### ผลลัพธ์ที่ได้

ฟังก์ชันนี้จะทำการลบเรคคอร์ดในไฟล์ตาม ID ที่ระบุ หาก ID ไม่พบ ฟังก์ชันจะแสดงข้อความแจ้งเตือน แต่ยังคงลบเรคคอร์ดที่มี ID ตรงกับที่ระบุไว้ต่อไป

### ตัวอย่างการเรียกใช้

ตัวอย่างนี้จะทำการลบเรคคอร์ดที่มี ID "001", "003", และ "005" จากไฟล์ไบนารี หากพบว่า ID ใดไม่อยู่ในไฟล์ ฟังก์ชันจะแจ้งข้อความว่าหาไม่พบ: deleteData(["001", "003", "005"])

## การสร้างรายงานสรุปข้อมูลและบันทึกเป็นไฟล์ข้อความ

ฟังก์ชันนี้ชื่อว่า exportReport ทำหน้าที่สร้างรายงานสรุปผลการทำงานของพนักงานตามแผนก และบันทึกรายงานนี้ลงในไฟล์ข้อความชื่อ report.txt

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑39 การสร้างรายงานสรุปข้อมูลและบันทึกเป็นไฟล์ข้อความ

### การอ่านข้อมูลจากไฟล์

ฟังก์ชันเริ่มต้นด้วยการอ่านข้อมูลทั้งหมดจากไฟล์ไบนารีโดยเรียกใช้ฟังก์ชัน readDataFromBinFile() และแปลงข้อมูลเป็นอาเรย์ของ NumPy โดยใช้ np.array() และกำหนดชนิดข้อมูลเป็น dtype="object" เพื่อให้รองรับข้อมูลประเภทต่าง ๆ ที่เก็บอยู่ในไฟล์

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑40 การอ่านข้อมูลจากไฟล์โดยใช้ numpy

### การจัดกลุ่มตามแผนก

* + - 1. สร้างตัวแปร departments ในรูปแบบของดิกชันนารี เพื่อเก็บข้อมูลของพนักงานที่จัดกลุ่มตามแผนก
      2. วนลูปในแต่ละเรคคอร์ดใน data\_bin\_file เพื่อดึงชื่อ (name), แผนก (dept), และคะแนนเฉลี่ย (avg\_score) ของพนักงานแต่ละคน โดยคำนวณคะแนนเฉลี่ยจากลิสต์ scores:

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑41 วนลูปในแต่ละเรคคอร์ดใน data\_bin\_file เพื่อดึงข้อมูล

* + - * 1. ถ้าแผนกของพนักงานไม่อยู่ในดิกชันนารี departments ให้สร้างแผนกนั้นขึ้นมา และกำหนดค่าเริ่มต้น เช่น รายชื่อพนักงาน (employees), คะแนนรวม (total\_score), จำนวนพนักงาน (count), และพนักงานที่ได้คะแนนสูงสุด (top\_performer)

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑42 การสร้างแผนกของพนักงานไม่อยู่ในดิกชันนารี departments

* + - * 1. เพิ่มชื่อและคะแนนเฉลี่ยของพนักงานในรายการของแผนกนั้น

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑43 การเพิ่มชื่อและคะแนนเฉลี่ยของพนักงานในรายการของแผนก

* + - * 1. รวมคะแนนเฉลี่ยของพนักงานเข้าในคะแนนรวม (total\_score) และเพิ่มจำนวนพนักงานในแผนก

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑44 การรวมคะแนนเฉลี่ยของพนักงานเข้าและเพิ่มจำนวนพนักงานในแผนก

* + - * 1. ถ้าคะแนนเฉลี่ยของพนักงานคนนี้มากกว่าคะแนนเฉลี่ยสูงสุดเดิมของแผนก ฟังก์ชันจะบันทึกให้พนักงานคนนี้เป็นผู้ทำคะแนนสูงสุดในแผนก (top\_performer)



ภาพที่ 3‑45 การบันทึกพนักงานที่ทำคะแนนสูงสุดในแผนก

### การหาพนักงานของแผนกที่ดีที่สุด

* + - 1. หลังจากวนลูปเพื่อเก็บข้อมูลทุกแผนกแล้ว ฟังก์ชันจะเริ่มต้นหาผลงานของแผนกที่ดีที่สุด โดยคำนวณคะแนนเฉลี่ยของแผนก (avg\_dept\_score) จาก total\_score หารด้วยจำนวนพนักงาน (count)

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑46 การหาพนักงานของแผนกที่ดีที่สุด

* + - 1. ถ้าคะแนนเฉลี่ยของแผนกใดมากกว่าคะแนนเฉลี่ยสูงสุดเดิม ฟังก์ชันจะบันทึกแผนกนั้นเป็นแผนกที่ดีที่สุด

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑47 คะแนนเฉลี่ยของแผนกใดมากกว่าคะแนนเฉลี่ยสูงสุดเดิม

### การสร้างข้อความรายงาน

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑48 การสร้างข้อความรายงาน

* + - 1. ฟังก์ชันสร้างสตริงรายงาน (str\_report) โดยมีหัวข้อหลัก "Summary Report" และข้อมูลรายละเอียดของแต่ละแผนก:
         1. แสดงชื่อแผนก และข้อมูลพนักงานแต่ละคนในแผนกพร้อมคะแนนเฉลี่ย (Average Score)
         2. แสดงพนักงานที่ทำคะแนนสูงสุดในแผนก (Top Performer)
      2. สุดท้าย ฟังก์ชันจะแสดงชื่อแผนกที่มีคะแนนเฉลี่ยดีที่สุด (Best Department) พร้อมกับคะแนนเฉลี่ยของแผนกนั้น

### การบันทึกลงไฟล์

ฟังก์ชันบันทึกสตริงรายงานลงในไฟล์ข้อความชื่อ report.txt โดยใช้ with open("report.txt", "w") ซึ่งจะเปิดไฟล์ในโหมดเขียน (ถ้าไฟล์มีอยู่แล้วจะถูกเขียนทับ)

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑49 บันทึกสตริงรายงานลงในไฟล์ข้อความชื่อ report.txt

### การแสดงผล

ฟังก์ชันพิมพ์ข้อความว่า "Report exported to report.txt successfully." เมื่อการเขียนรายงานลงไฟล์สำเร็จ

### ผลลัพธ์ที่ได้

ฟังก์ชันนี้จะสร้างรายงานสรุปผลการทำงานของพนักงานในแต่ละแผนก โดยแสดงข้อมูลพนักงานในแผนก คะแนนเฉลี่ยของแต่ละคน และพนักงานที่ทำคะแนนสูงสุดในแผนกนั้น รวมถึงระบุแผนกที่มีคะแนนเฉลี่ยดีที่สุด และบันทึกรายงานทั้งหมดลงในไฟล์ report.txt

### ตัวอย่างการเรียกใช้

เมื่อเรียกฟังก์ชันนี้ จะสร้างรายงานสรุปผลของพนักงานทุกคนตามแผนกและบันทึกลงไฟล์: exportReport()

## แสดงข้อมูลของพนักงานทั้งหมด

ฟังก์ชันนี้ชื่อว่า showAllData ทำหน้าที่แสดงข้อมูลพนักงานทั้งหมดจากไฟล์ไบนารี ซึ่งสามารถเลือกได้ว่าจะให้แสดงเฉลี่ยคะแนนหรือแสดงคะแนนแบบเต็ม

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑50 การแสดงข้อมูลของพนักงานทั้งหมด

### การอ่านข้อมูลจากไฟล์

ฟังก์ชันเริ่มต้นด้วยการเรียกใช้ฟังก์ชัน readDataFromBinFile() เพื่ออ่านข้อมูลทั้งหมดจากไฟล์ไบนารี และเก็บข้อมูลไว้ในตัวแปร list\_records ซึ่งเป็นลิสต์ของเรคคอร์ด

### การเลือกโหมดการแสดงข้อมูล

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑51 การเลือกโหมดการแสดงข้อมูล

ฟังก์ชันรับพารามิเตอร์ choice เพื่อระบุว่าจะแสดงข้อมูลในรูปแบบใด:

* + - 1. ถ้า choice คือ '1' ฟังก์ชันจะแสดงข้อมูลพร้อมคะแนนเฉลี่ยของแต่ละพนักงาน
      2. ถ้า choice คือ '2' ฟังก์ชันจะแสดงข้อมูลพร้อมคะแนนของแต่ละพนักงานในรูปแบบลิสต์ที่คั่นด้วยเครื่องหมายจุลภาค (,)

### การแสดงข้อมูลในกรณี choice = '1' (แสดงคะแนนเฉลี่ย)

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑52 การแสดงข้อมูลในกรณี choice = '1' (แสดงคะแนนเฉลี่ย)

* + - 1. ฟังก์ชันจะสร้างลิสต์ใหม่จาก list\_records โดยแทนที่ข้อมูลคะแนนของแต่ละพนักงาน (record[3]) ด้วยค่าเฉลี่ยของคะแนนเหล่านั้นโดยใช้ np.mean()
      2. กำหนดชนิดข้อมูลของลิสต์นี้ในรูปแบบ dtype เพื่อให้รองรับฟิลด์ที่ต้องการ ได้แก่:
         1. ID (ประเภท U20 สำหรับตัวอักษร Unicode ไม่เกิน 20 ตัว)
         2. Name (ประเภท U20)
         3. Department (ประเภท U20)
         4. Average Score (คะแนนเฉลี่ย ประเภท f4 หรือจำนวนทศนิยมแบบ float)
         5. Salary (ประเภท f4)
      3. จากนั้นแปลงข้อมูลในลิสต์เป็นอาเรย์ NumPy (np.asarray()) เพื่อใช้จัดการต่อไป

### การแสดงข้อมูลในกรณี choice = '2' (แสดงคะแนนแบบแจกแจ้ง)

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑53 การแสดงข้อมูลในกรณี choice = '2' (แสดงคะแนนแบบแจกแจ้ง)

* + - 1. ฟังก์ชันจะแปลงลิสต์ของข้อมูลพนักงาน โดยใช้ ', '.join() เพื่อรวมคะแนนแต่ละค่าที่อยู่ใน record[3] เป็นสตริงคั่นด้วยเครื่องหมายจุลภาค
      2. กำหนดชนิดข้อมูล dtype ในรูปแบบที่คล้ายกับตัวเลือกแรก แต่ใช้ฟิลด์ Score (ประเภท U20) เพื่อแทนที่ฟิลด์ Average Score

### การตรวจสอบและเรียงลำดับข้อมูล

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑54 การตรวจสอบและเรียงลำดับข้อมูล

* + - 1. ฟังก์ชันจัดเรียงข้อมูลในอาเรย์โดยเรียงตามฟิลด์ ID โดยใช้ np.sort(data\_arr, order='ID')
      2. แปลงอาเรย์ที่จัดเรียงแล้วเป็น DataFrame ของ Pandas (pd.DataFrame  
         (sorted\_data)) เพื่อให้ง่ายต่อการจัดการและแสดงผล

### การตรวจสอบข้อมูล

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑55 การตรวจสอบข้อมูล

* + - 1. ถ้า DataFrame ว่างเปล่า (ไม่มีข้อมูล) ฟังก์ชันจะแสดงข้อความว่า "Data not found."
      2. ถ้ามีข้อมูล ฟังก์ชันจะแสดงข้อมูลทั้งหมดในรูปแบบตารางโดยใช้คำสั่ง print(df\_data\_arr.to\_string(index=False)) ซึ่งจะพิมพ์ข้อมูลในรูปแบบที่ไม่มีตัวเลขลำดับ (index=False)

### ผลลัพธ์ที่ได้

ฟังก์ชันนี้จะพิมพ์ข้อมูลพนักงานทั้งหมดในรูปแบบตาราง โดยขึ้นอยู่กับตัวเลือก choice:

* + - 1. ถ้าเลือก '1' จะแสดง ID, ชื่อ, แผนก, คะแนนเฉลี่ย, และเงินเดือน จากภาพที่ 2-3
      2. ถ้าเลือก '2' จะแสดง ID, ชื่อ, แผนก, คะแนนเต็มของแต่ละพนักงาน, และเงินเดือน จากภาพที่ 2-4

### ตัวอย่างการเรียกใช้

ฟังก์ชันนี้สามารถเรียกใช้ได้โดยกำหนดตัวเลือกการแสดงข้อมูล เช่น:

* + - 1. showAllData('1') เป็นการแสดงข้อมูลพร้อมคะแนนเฉลี่ย
      2. showAllData('2') เป็นการแสดงข้อมูลพร้อมคะแนนเต็ม

## แสดงข้อมูลเฉพาะของพนักงานตามคอลัมน์ที่กำหนด

ฟังก์ชันนี้ชื่อว่า showSpecificData ทำหน้าที่แสดงข้อมูลเฉพาะของพนักงานตามคอลัมน์ที่กำหนด โดยสามารถกรองข้อมูลตามเงื่อนไขที่ต้องการ เช่น การค้นหาข้อมูลจากคอลัมน์ ID, Name, Department, Average Score, Score หรือ Salary

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑56 การแสดงข้อมูลเฉพาะของพนักงานตามคอลัมน์ที่กำหนด

### รายละเอียดการทำงาน

ฟังก์ชันนี้รับพารามิเตอร์ 3 ตัว ได้แก่:

* + - 1. col: หมายเลขคอลัมน์ที่ต้องการค้นหา เช่น 1 (ID), 2 (Name), 3 (Department), 4 (Average Score), 5 (Score), และ 6 (Salary)
      2. list\_search: ลิสต์ของค่าที่ต้องการใช้ในการค้นหา
      3. choice: เงื่อนไขการเปรียบเทียบค่าต่างๆ เช่น มากกว่า, น้อยกว่า, หรือเท่ากับ

### การอ่านข้อมูลจากไฟล์

ฟังก์ชันเริ่มต้นด้วยการเรียกใช้ฟังก์ชัน readDataFromBinFile() เพื่ออ่านข้อมูลทั้งหมดจากไฟล์ไบนารี และเก็บข้อมูลไว้ในตัวแปร list\_records ซึ่งเป็นลิสต์ของเรคคอร์ด

### การกำหนดคอลัมน์ที่ต้องการค้นหา

A screen shot of a computer program

Description automatically generated

ภาพที่ 3‑57 การกำหนดคอลัมน์ที่ต้องการค้นหา

* + - 1. ฟังก์ชันจะแปลงค่าพารามิเตอร์ col เป็นชื่อของคอลัมน์ เช่น 'ID', 'Name', 'Department', 'Average Score', 'Score', หรือ 'Salary' โดยการแมปหมายเลขคอลัมน์เข้ากับชื่อคอลัมน์ในลิสต์ที่กำหนดไว้ล่วงหน้า
      2. จากนั้นฟังก์ชันจะแบ่งการจัดการข้อมูลเป็นสองรูปแบบ ขึ้นอยู่กับคอลัมน์ที่เลือก

### การจัดการข้อมูลในคอลัมน์ Score

ถ้าเลือกค้นหาคอลัมน์ Score ฟังก์ชันจะเปลี่ยน record[3] ซึ่งเป็นลิสต์ของคะแนน เป็นสตริงที่คั่นด้วยเครื่องหมายจุลภาค (,) และเก็บข้อมูลเป็นอาเรย์ NumPy พร้อมชนิดข้อมูล dtype ที่ประกอบไปด้วยฟิลด์ ID, Name, Department, Score, และ Salary

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑58 การจัดการข้อมูลในคอลัมน์ Score

### การจัดการข้อมูลในคอลัมน์อื่นๆ

ถ้าเลือกค้นหาคอลัมน์ที่ไม่ใช่ Score เช่น Average Score หรือ Salary ฟังก์ชันจะคำนวณค่าเฉลี่ยของคะแนนแต่ละพนักงาน (record[3]) โดยใช้ np.mean() และเก็บข้อมูลในอาเรย์ NumPy พร้อมชนิดข้อมูล dtype ที่ประกอบไปด้วยฟิลด์ ID, Name, Department, Average Score, และ Salary

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑59 การจัดการข้อมูลในคอลัมน์อื่นๆ

### การเปรียบเทียบและกรองข้อมูล

* + - 1. ถ้าค้นหาคอลัมน์ที่เป็นตัวเลข เช่น Average Score หรือ Salary ฟังก์ชันจะใช้พารามิเตอร์ choice เพื่อกำหนดเงื่อนไขการกรองข้อมูล เช่น มากกว่า (>) หรือ น้อยกว่า (<) ตามค่าที่ผู้ใช้ระบุใน list\_search
         1. '1': มากกว่า (>)
         2. '2': มากกว่าหรือเท่ากับ (>=)
         3. '3': น้อยกว่า (<)
         4. '4': น้อยกว่าหรือเท่ากับ (<=)
         5. ถ้าไม่ได้เลือกเงื่อนไขเฉพาะ ฟังก์ชันจะใช้ค่าใน list\_search เพื่อกรองข้อมูลที่ตรงกับค่าที่ค้นหา

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑60 ค้นหาคอลัมน์ที่เป็นตัวเลข เช่น Average Score หรือ Salary

* + - 1. ถ้าค้นหาคอลัมน์ที่เป็นสตริง เช่น ID, Name, หรือ Department ฟังก์ชันจะใช้ np.isin() เพื่อกรองข้อมูลที่มีค่าในคอลัมน์เหล่านั้นตรงกับค่าใน list\_search

A screen shot of a computer code

Description automatically generated

ภาพที่ 3‑61 ค้นหาคอลัมน์ที่เป็นสตริง เช่น ID, Name, หรือ Department

### การเรียงลำดับข้อมูล

ข้อมูลที่ถูกกรองแล้วจะถูกจัดเรียงตามลำดับตามคอลัมน์ที่เกี่ยวข้อง:

* + - 1. ถ้าคอลัมน์ที่ค้นหาคือ ID หรือ Name ข้อมูลจะถูกจัดเรียงตาม ID ตามภาพที่ 3-61
      2. ถ้าคอลัมน์ที่ค้นหาคือ Department ข้อมูลจะถูกจัดเรียงตาม Department ตามภาพที่ 3-61
      3. ถ้าคอลัมน์ที่ค้นหาคือ Score ข้อมูลจะถูกจัดเรียงตาม ID ตามภาพที่ 3-61
      4. ถ้าคอลัมน์ที่ค้นหาคือ Average Score หรือ Salary ข้อมูลจะถูกจัดเรียงตามคอลัมน์ที่ค้นหานั้น ตามภาพที่ 3-60

### การแสดงผล

* + - 1. ฟังก์ชันจะนำข้อมูลที่ถูกจัดเรียงและกรองแล้วมาแปลงเป็น DataFrame ของ Pandas (pd.DataFrame(data=sorted\_data)) เพื่อให้สามารถแสดงข้อมูลในรูปแบบตาราง
      2. ถ้า DataFrame ว่างเปล่า (ไม่มีข้อมูลตรงตามที่ค้นหา) ฟังก์ชันจะแสดงข้อความว่า "No data found."
      3. ถ้ามีข้อมูล ฟังก์ชันจะแสดงข้อมูลทั้งหมดในรูปแบบตารางโดยใช้คำสั่ง print(df\_data\_fltr.to\_string(index=False)) ซึ่งจะพิมพ์ข้อมูลในรูปแบบที่ไม่มีตัวเลขลำดับ (index=False)

### ผลลัพธ์ที่ได้

ฟังก์ชันนี้จะกรองและแสดงข้อมูลพนักงานเฉพาะที่ตรงกับเงื่อนไขที่ผู้ใช้กำหนด เช่น ค่าตัวเลขที่มากกว่าหรือน้อยกว่า หรือตรงกับค่าที่ค้นหาในคอลัมน์ที่ต้องการ

### ตัวอย่างการเรียกใช้

ฟังก์ชันนี้สามารถเรียกใช้โดยกำหนดคอลัมน์และเงื่อนไขการค้นหา เช่น:

* + - 1. showSpecificData(1, ['12345']) เป็นการค้นหาข้อมูลจากคอลัมน์ ID
      2. showSpecificData(4, [80], '1') เป็นการค้นหาข้อมูลที่มีคะแนนเฉลี่ยมากกว่า 80

# อธิบายการทำงานของ code ส่วนการทำงาน

## การรับคำสั่งจากผู้ใช้ (User Input Handling)

โค้ดที่ให้ไปเริ่มต้นด้วยการแสดงเมนูหลักให้ผู้ใช้เลือกการทำงานต่างๆ ซึ่งแต่ละเมนูจะสอดคล้องกับฟังก์ชันเฉพาะที่รับผิดชอบการทำงาน เช่น:

1. แสดงข้อมูลทั้งหมด (Show all data): ใช้สำหรับแสดงข้อมูลพนักงานทั้งหมด โดยสามารถเลือกรูปแบบการแสดงข้อมูลได้
2. แสดงข้อมูลเฉพาะเจาะจง (Show specific data): ใช้สำหรับค้นหาข้อมูลพนักงานตามเงื่อนไขที่ระบุ เช่น ค้นหาจาก ID, ชื่อ, แผนก หรือคะแนนเฉลี่ย
3. เพิ่มข้อมูล (Insert data): ใช้สำหรับเพิ่มข้อมูลพนักงานใหม่
4. แก้ไขข้อมูล (Edit data): ใช้สำหรับแก้ไขข้อมูลพนักงานที่มีอยู่แล้ว
5. ลบข้อมูล (Delete data): ใช้สำหรับลบข้อมูลพนักงานที่ระบุ
6. สร้างรายงาน (Export report): ใช้สำหรับสร้างรายงานสรุปข้อมูลพนักงาน

## โครงสร้างการทำงานของแต่ละเมนู (Menu Functionality)

1. แสดงข้อมูลทั้งหมด (Show all data): โค้ดจะเรียกใช้ฟังก์ชัน showAllData ซึ่งจะรับพารามิเตอร์จากผู้ใช้เพื่อกำหนดรูปแบบการแสดงข้อมูลที่ต้องการ เช่น แสดงเฉลี่ยคะแนนหรือแสดงคะแนนล่าสุด
2. แสดงข้อมูลเฉพาะเจาะจง (Show specific data): ฟังก์ชัน showSpecificData จะรับพารามิเตอร์เพื่อทำการค้นหาข้อมูลเฉพาะตามที่ผู้ใช้ระบุ โค้ดจะเรียงลำดับข้อมูลที่ตรงกับเงื่อนไขก่อนแสดงผลออกมา
3. เพิ่มข้อมูล (Insert data): ฟังก์ชันนี้จะรับข้อมูลพนักงานใหม่จากผู้ใช้ เช่น ID, ชื่อ, แผนก, คะแนน, และเงินเดือน แล้วทำการเพิ่มข้อมูลใหม่ลงในไฟล์ข้อมูล
4. แก้ไขข้อมูล (Edit data): ผู้ใช้สามารถเลือกแก้ไขข้อมูลที่ต้องการ เช่น ชื่อ, แผนก, คะแนน หรือเงินเดือน โดยระบุ ID ของพนักงานที่ต้องการแก้ไข
5. ลบข้อมูล (Delete data): โค้ดจะรับ ID ของพนักงานที่ต้องการลบ แล้วทำการลบข้อมูลนั้นจากไฟล์
6. สร้างรายงาน (Export report): ฟังก์ชันนี้จะสร้างรายงานข้อมูลพนักงานในรูปแบบที่กำหนด

## การจัดการข้อผิดพลาด (Error Handling)

ในแต่ละส่วนของโค้ดได้มีการจัดการกับข้อผิดพลาดที่อาจเกิดขึ้น โดยใช้ try-except เพื่อดักจับข้อผิดพลาดที่อาจเกิดขึ้นในระหว่างการทำงาน เช่น การป้อนข้อมูลผิดพลาด การไม่พบข้อมูล หรือการยกเลิกการทำงาน (เช่น การกด Ctrl+C)

## โครงสร้างข้อมูลและการจัดเก็บ (Data Structure and Storage)

ข้อมูลของพนักงานถูกจัดเก็บในไฟล์ไบนารีและเรียกใช้ผ่านโมดูล BinFileOperation ซึ่งเป็นการทำงานที่เกี่ยวข้องกับการอ่านและเขียนข้อมูลในรูปแบบไบนารีเพื่อประสิทธิภาพในการจัดเก็บข้อมูล โดย ข้อมูลที่จัดเก็บจะประกอบด้วย ID, ชื่อ, แผนก, คะแนน และเงินเดือน โดยจะใช้โครงสร้างแบบ numpy และ pandas ในการจัดการข้อมูลเพื่อความสะดวกในการค้นหาและแสดงผล

## การใช้โมดูลเสริม (External Libraries)

ในโค้ดนี้มีการใช้โมดูล numpy และ pandas เพื่อช่วยในการจัดการข้อมูลที่เป็นตัวเลขและข้อความได้อย่างมีประสิทธิภาพ โดย numpy ช่วยในการสร้างอาร์เรย์ข้อมูลและการประมวลผลเชิงตัวเลข ส่วน pandas ช่วยในการสร้างตารางข้อมูล (DataFrame) เพื่อแสดงผลในรูปแบบที่อ่านง่าย

## สรุป (Conclusion)

รายงานสามารถสรุปได้ว่าโปรแกรมนี้เป็นเครื่องมือที่มีประสิทธิภาพในการจัดการข้อมูลพนักงาน โดยใช้การป้อนข้อมูลจากผู้ใช้เพื่อตอบสนองการทำงานต่างๆ ทั้งการเพิ่ม ลบ แก้ไข และค้นหาข้อมูล อีกทั้งยังสามารถสร้างรายงานได้อย่างสะดวก โปรแกรมนี้เหมาะสมกับการจัดการข้อมูลในองค์กรหรือหน่วยงานที่ต้องการการบันทึกข้อมูลพนักงานในลักษณะที่เป็นระบบ

A screenshot of a computer program

Description automatically generated

ภาพที่ 4‑1 โค้ดแสดงเมนูหลัก

A screenshot of a computer screen

Description automatically generated

ภาพที่ 4‑2 โค้ดแสดงเมนูของการแสดงข้อมูลทั้งหมด

A screen shot of a computer program

Description automatically generated

ภาพที่ 4‑3 โค้ดแสดงเมนูของการแสดงผลแบบกรองข้อมูล

A screen shot of a computer code

Description automatically generated

ภาพที่ 4‑4 โค้ดแสดงเมนูของการเพิ่มข้อมูล

A screen shot of a computer code

Description automatically generated

ภาพที่ 4‑5 รับคำสั่งจากผู้ใช้สำหรับการแก้ไขข้อมูลพนักงานตามคอลัมน์ที่เลือกและตาม ID ที่ป้อน

A black background with text

Description automatically generated

ภาพที่ 4‑6 การรับชื่อใหม่ที่ผู้ใช้ป้อนสำหรับพนักงานตาม ID ที่ต้องการแก้ไข

A black screen with text

Description automatically generated with medium confidence

ภาพที่ 4‑7 การแก้ไขข้อมูลแผนกของพนักงานตาม ID ที่ผู้ใช้เลือก

A screen shot of a computer code

Description automatically generated

ภาพที่ 4‑8 การแก้ไขคะแนน (Score) ของผู้ใช้

A screen shot of a computer program

Description automatically generated

ภาพที่ 4‑9 การทำงานย่อยที่ 1 ของการแก้ไขคะแนน (score)

A screen shot of a computer program

Description automatically generated

ภาพที่ 4‑10 การทำงานย่อยที่ 2 ของการแก้ไขคะแนน (score)

A screenshot of a computer program

Description automatically generated

ภาพที่ 4‑11 การเพิ่ม, ตัด, และแก้ไขเงินเดือนและแสดงผลคำแนะนำเพื่อให้ผู้ใช้ป้อนข้อมูล

A computer screen shot of a computer code

Description automatically generated

ภาพที่ 4‑12 การรับค่า ID (หรือหลาย ID) จากผู้ใช้เพื่อลบข้อมูลที่มี ID เหล่านั้น

A screenshot of a computer screen

Description automatically generated

ภาพที่ 4‑13 การทำรายงานและการแสดงข้อความยืนยันออกจากโปรแกรม

A screen shot of a computer program

Description automatically generated

ภาพที่ 4‑14 การจัดการข้อผิดพลาดที่อาจเกิดขึ้นในโปรแกรม