ใบงานที่ 23

เรื่อง ต้นไม้เพื่อการค้นหา

เสนอ

อาจารย์ ปิยพล ยืนยงสถาวร

จัดทำโดย

นาย สารินทร์ อินต๊ะรักษา รหัส 65543206082-1

ใบงานนี้เป็นส่วนหนึ่งของรายวิชา โครงสร้างข้อมูลและขั้นตอนวิธี

หลักสูตรวิศวกรรมคอมพิวเตอร์ คณะวิศวกรรมศาสตร์

มหาวิทยาลัยเทคโนโลยีราชมงคลล้านนา

ประจำภาคที่ 1 ปีการศึกษา 2566

คำสั่ง/คำชี้แจง

- สร้างโค้ดโปรแกรมตามตัวอย่างในเอกสารประกอบการสอน
- แสดงโค้ดโปรแกรมเป็นส่วนๆ พร้อมอธิบาย
- แสดงผลการรันโปรแกรม พร้อมอธิบายการทำงาน
- สรุปผลการทดลอง

ลำดับขั้นการทดลอง

```
int Data[MaxData];
int N,key,Times;
bool result;
bool Duplicate(int i,int Data1) //Check Duplication Data
{
int j;
for(j=1;j<=i;j++)
{
if(Data1==Data[j])
return(true);
}
return(false);
}
void PrepareRawKey(int N)
{
int i,j,temp;
srand(time(NULL)); //for difference random number in rand()
for (i=1;i<=N;i++)
{
temp=(rand() % 89)+10; //random difference number 10..99
while(Duplicate(i-1,temp)) //Loop if Still Duplicate
temp=(rand() % 89)+10; //random again
Data[i]=temp; //Keep new Number
} //End for
} //End Fn.
```

- ประกาศตัวแปรที่ใช้ในโปรแกรม, รวมถึงอาร์เรย์ Data ที่จะใช้ในการเก็บข้อมูล
- bool Duplicate คือฟังก์ชันที่ใช้ในการตรวจสอบว่าค่า Data1 มีค่าเดียวกันกับข้อมูลในอาร์เรย์ Data ถ้ามีฟังก์ชันจะคืนค่า true ถ้าไม่มีจะคืนค่า false
- void PrepareRawKey คือฟังก์ชันที่ใช้ในการเตรียมข้อมูลสุ่ม N ค่าและเก็บในอาร์เรย์ Data
- ใช้ srand เพื่อกำหนดค่าเริ่มต้นสำหรับการสร้างเลขสุ่ม
- ในลูป for จะสร้างค่าสุ่ม temp ในช่วง 10 ถึง 99
- จากนั้นจะใช้ฟังก์ชัน Duplicate เพื่อตรวจสอบว่าค่า temp ซ้ำกับข้อมูลในช่วงที่สร้างข้อมูลไปแล้วหรือไม่
- หากค่า temp ซ้ำกับข้อมูลที่มีอยู่แล้วในอาร์เรย์ Data จะทำการสร้างค่าสุ่มใหม่

```
void DispKey(int N)
{
int i;
for(i=1;i<=N;i++)
printf("(%2d)",i); //Show Subscript i
printf("\n");
for(i=1;i<=N;i++)
printf(" %2d ",Data[i]); //Show Data[]
printf("\n");
}
void BubbleSort(int N) //Ascending Sort
{
int i,j,temp;
for(i=1;i<=N-1;i++) //loop forward
{
if(Data[i]>Data[i+1]) //if not true position
{
j=i+1; //loop backward
while(Data[j]<Data[j-1]) //while if remain bubble
{
temp=Data[j-1]; //swap data
Data[j-1]=Data[j];
Data[j]=temp;
j--; //count down j
} //end while
} //end if
} //end for
} //end Fn.
```

- void DispKey ฟังก์ชันนี้ใช้ในการแสดงข้อมูลที่อยู่ในอาร์เรย์ Data ที่มีขนาด N และแสดงข้อมูลทั้งแบบแสดงหมายเลขลำดับ (subscript) และข้อมูลจริงของอาร์เรย์ Data.
- void BubbleSort ฟังก์ชันนี้ใช้ในการเรียงลำดับข้อมูลในอาร์เรย์ Data โดยใช้วิธี Bubble Sort ในลำดับน้อยไปมาก (ascending order)

```
bool BinarySearch(int Key1)
{
int L,R,Mid;
L=1;
R=N;
Times=0; //Initial Time for search
while(L<=R)
{
Mid=(L+R)/2; //Calculate Middle
Times++; //Count Searching Time
printf("L : %2d ",L);
printf("R : %2d ",R);
printf("Mid : %2d ",Mid);
printf("Searching Time : %d\n",Times);
if(Key1==Data[Mid])
return(true); //if found
else
{
if(Key1<Data[Mid])
R=Mid-1; //Move R
else
L=Mid+1; //Move L
}
} //End while
return(false); //If not found
} //End Fn.
```

ในฟังก์ชัน BinarySearch กำหนดตัวแปรและการทำงานในแต่ละขั้นตอนของการค้นหาค่า Key1 ในอาร์เรย์ Data ที่เรียงลำดับแบบน้อยไปมาก (ascending order) อย่างละเอียดและชัดเจนโดยใช้ตัวแปร L, R, Mid, และ Times เพื่อนับจำนวนครั้งที่ค้นหาและแสดงข้อมูลที่เกี่ยวข้องกับการค้นหาบนหน้าจอ การค้นหาจะสิ้นสุดเมื่อ L มีค่ามากกว่า R และคืนค่า true ถ้าค่า Key1 พบในอาร์เรย์ Data, มิฉะนั้นคืนค่า false ถ้าไม่พบ

```
int main()
{
printf("BINARY SEARCH\n");
printf("================================================\n");
N=32;
PrepareRawKey(N);
printf("Raw key :\n");
DispKey(N); //Raw key
BubbleSort(N);
printf("-----------------------------------------------------------------\n");
while(key!=-999)
{
printf("Sorted Key :\n");
DispKey(N); //Sorted Key
printf("\nEnter Key for Search(-999 for EXIT) = ");
scanf("%d",&key); //Read key from KBD
if(key!=-999)
{
result=BinarySearch(key); //Call Binary Search
if(result)
printf("Result...FOUND\n"); //if found
else
{
Beep(600,600);
printf("Result...NOT FOUND!!\n"); //if NOT found
}
//printf("Searching Time : %d\n",Times);
printf("-------------------------------------------------------Searching Finished\n");
} //End if
} //End While
return(0);
} //End Main
```

- แสดงข้อความ "BINARY SEARCH"
- กำหนดค่าตัวแปร N เป็น 32 และเตรียมข้อมูลสำหรับค้นหาโดยเรียกใช้ฟังก์ชัน PrepareRawKey
- แสดงข้อมูลเริ่มต้นของอาร์เรย์ "Raw key"
- เรียกใช้ฟังก์ชัน BubbleSort(N) เพื่อเรียงลำดับข้อมูลในอาร์เรย์ Data
- ลูป while ที่ทำงานเมื่อ key ไม่เท่ากับ -999.
- แสดงข้อมูลที่เรียงลำดับแล้วของอาร์เรย์ "Sorted Key"
- รับค่า key จากผู้ใช้โดยใช้ scanf.
- ถ้า key ไม่เท่ากับ -999, จะเรียกใช้ฟังก์ชัน BinarySearch(key) เพื่อค้นหาค่า key ในอาร์เรย์ Data.
- หากค่า key พบในอาร์เรย์ Data, จะแสดงข้อความ "Result...FOUND" และถ้าไม่พบจะแสดง "Result...NOT FOUND!!" และใช้ Beep เพื่อสร้างเสียงเพื่อแจ้งเตือน
- ลูปจะทำงานไปเรื่อยๆ จนกว่าผู้ใช้จะป้อนค่า -999 เพื่อออกจากโปรแกรม.

ผลลัพธ์การทดลอง

สรุปผลการทดลอง

การทดลองนี้เป็นการใช้โปรแกรมในการค้นหาค่าในอาร์เรย์ที่เรียงลำดับแบบน้อยไปมา (ascending order) โดยใช้วิธีการค้นหาแบบทวิภาค (Binary Search) และเรียงลำดับข้อมูลด้วย Bubble Sort ก่อนการค้นหา

สื่อ / เอกสารอ้างอิง

ไฟล์ประกอบการสอนของ อาจารย์ ปิยพล ยืนยงสถาวร เรื่อง : ต้นไม้เพื่อการค้นหา