Tarea unidad 5: Métodos de recuperación de información

Diana Laura Reyes Youshimatz ID: 173391

Problema 1. Búsqueda secuencial y binaria (3 puntos). Estudie y analice los algoritmos de los métodos de búsqueda de información. Desarrolle un programa en C basado en arreglos, que demuestre la dinámica de cada uno de los siguientes algoritmos: 1. Búsqueda secuencial (1 punto) 2. Búsqueda binaria (1 punto)

```
#include <stdio.h>

#include <stdlib.h>

#include <time.h>

void bubble_sort(int arr[],int n){

  int i,j,temp;

  for(i=0;i<n;i++){

     for(j=0;j<n-i-1;j++){

        if(arr[j]>arr[j+1]){

          temp=arr[j];

          arr[j]=arr[j+1];

          arr[j+1]=temp;

          }

       }

    }

}

int bsec(int arr[], int n, int x) {

  for (int i=0; i<n; i++)

    if (arr[i]==x)

      return i;

  return -1;

}

int bbin(int arr[], int l, int r, int x) {

    while (l<=r){

       int m=l+(r-l)/2;
```

```
        if (arr[m]==x)
            return m;
        if (arr[m]<x)
            l= m+1;
        else
            r= m-1;
    }
    return -1;
}

int main() {
  printf("Size del arreglo: ");
  int n; scanf("%d",&n);
  int x = 1; int i, choice;
  printf("\nArreglo: ");
  int arr[n];
  srand(time(NULL));
            for (i = 0; i<n; i++) {
                arr[i] = rand()%100;
            printf(" %d ", arr[i]);
  }
  printf("\n        1.Busqueda secuencial \n          2.Busqueda binaria \n   3. Salir ");
  while(1){
            printf("\nElemento a buscar: "); scanf("%d",&x);
            printf("\nIngresa el numero de la opcion: "); scanf("%d",&choice);


    switch(choice){
            case 1:
```

```
                    (bsec(arr, n, x)== -1) ? printf("Error. Not found") : printf("Se encuentra en
posicion: %d",bsec(arr, n, x));
                    break;
            case 2:
                    bubble_sort(arr,n);
                     printf("\nArreglo ordenado: ");
                            for (i = 0; i<n; i++) {
                            printf(" %d ", arr[i]);
                     }
                    printf("\n");
                    (bbin(arr,0,n-1,x) == -1) ? printf("Error. Not found") : printf("Se encuentra
en posicion: %d", bbin(arr,0,n-1,x));
                    break;
            case 3:
                    exit(0);
        }
  }
}
```

Problema 2. Hashing (3 puntos) Estudie y analice el algoritmo de búsqueda Hashing. Desarrolle un programa en C que demuestre la inserción y búsqueda de datos en un arreglo usando Hashing.

```
#include <stdio.h>
#include <stdlib.h>
#define TABLE_SIZE 9
#define Prim 6

class Hashdoble
{
   int *tabla;
   int tam;
public:
```

```
bool full(){
   return (tam == TABLE_SIZE);
}
int hash1(int key){
   return (key%TABLE_SIZE);
}
int hash2(int key){
   return (Prim-(key%Prim));
}
Hashdoble(){
   tabla = new int[TABLE_SIZE];
   tam = 0;
   for (int i=0; i<TABLE_SIZE; i++)
      tabla[i] =-1;
}
void insert(int key) {
   if (full())
      return;
   int i=hash1(key);

   if (tabla[i]!=-1) {
      int i2=hash2(key);
      int i=1;
      while (1){
         int newi=(i+i*i2)%TABLE_SIZE;
         if (tabla[newi] == -1){
            tabla[newi] = key;
            break;
         }
```

```
            i++;
          }
       }
       else
          tabla[i] = key;
       tam++;
    }
    void print(){
       for (int i = 0; i < TABLE_SIZE; i++){
          if (tabla[i] != -1)
             printf(" %d --->%d\n",i,tabla[i]);
          else
            printf(" %d \n",i);

       }
    }
};

int main()
{
    int a[] = {4, 17, 30, 55, 90, 11, 56, 77};
    int n = sizeof(a)/sizeof(a[0]);
    Hashdoble h;
    for (int i = 0; i < n; i++)
       h.insert(a[i]);

    // display the hash Table
    h.print();
    return 0;
```

```
}
```