**M. Ali. Arslan**

**19F-0348**

**Lab # 13**

**Task # 01**

**Linear Probing**

#include <iostream>

#include <string>

using namespace std;

struct node

{

string price;

string name;

float Book\_id;

float quantity;

};

node arr[50];

void initialize();

int insert(node, int, int);

int hash\_fun(int, int);

void display(int);

void search(int);

void delete\_(int);

int main()

{

int choose, Size;

initialize();

while (1)

{

system("cls");

cout << "1) Insert a Book Data\n2) Display the List\n3) Search a Book\n4) Delete a Book\n5) Exit your Program\n\n";

cin >> choose;

cin.ignore();

node temp;

switch (choose)

{

case 1:

cout << "Enter Book Name: ";

getline(cin, temp.name);

cout << "Enter Book\_id: ";

cin >> temp.Book\_id;

cin.ignore();

cout << "Enter price: ";

getline(cin, temp.price);

Size = insert(temp, 0, 0);

if (Size == 50)

cout << "List is Full. New Number can't be Inserted...\n";

else

cout << "\nnumber inserted\n";

cout << endl << endl;

break;

case 2:

display(-1);

cout << "\n";

break;

case 3:

cout << "Enter Book\_id to Search: ";

cin >> choose;

cin.ignore();

search(choose);

break;

case 4:

cout << "Enter Book\_id to Delete: ";

cin >> choose;

cin.ignore();

delete\_(choose);

break;

}

cout << endl << endl << "Press Enter to Continue....";

getchar();

}

return 0;

}

void initialize()

{

for (int i = 0; i < 50; i++)

arr[i].Book\_id = -1;

}

int insert(node num, int f, int Size)

{

if (Size < 10)

{

Size++;

int index = hash\_fun(num.Book\_id, f);

if (arr[index].Book\_id == -1)

{

arr[index] = num;

}

else

return insert(num, ++f, Size);

}

return Size;

}

int hash\_fun(int num, int f)

{

return (num + f) % 10;

}

void search(int num)

{

int i = 0;

int index = hash\_fun(num, i++);

while ((arr[index].Book\_id != num) && (arr[index].Book\_id != -1) && (i < 20))

index = hash\_fun(num, i++);

if (arr[index].Book\_id == -1 || i > 50)

cout << "Book not found in the List\n";

else

{

cout << "\tBook with " << num << " Book\_id found at Index Number: " << index << "\n";

cout << "\nBook Name: " << arr[index].name << endl

<< "Book Book\_id: " << arr[index].Book\_id << endl

<< "Book price: " << arr[index].price << endl;

}

}

void delete\_(int num)

{

int i = 0;

int index = hash\_fun(num, i++);

while ((arr[index].Book\_id != num) && (arr[index].Book\_id != -1) && (i < 20))

index = hash\_fun(num, i++);

if (arr[index].Book\_id == -1 || i > 50)

cout << "Book Data not found in the list\n";

else

{

int zzz = arr[index].Book\_id;

arr[index].Book\_id = -1;

cout << "\tBook with " << num << " Book\_id has been deleted" << "\n";

cout << "\nBook Name: " << arr[index].name << endl

<< "Book Book\_id: " << zzz << endl

<< "Book price: " << arr[index].price << endl;

}

}

void display(int index = -1)

{

cout << "Books Database: \n\n";

for (int i = 0; i < 50; i++)

{

cout << "Index: " << i << "\t\t";

if (arr[i].Book\_id == -1)

cout << arr[i].Book\_id;

else

{

cout << "Book Name: " << arr[i].name << endl

<< "\t\t\tBook Book\_id: " << arr[i].Book\_id << endl

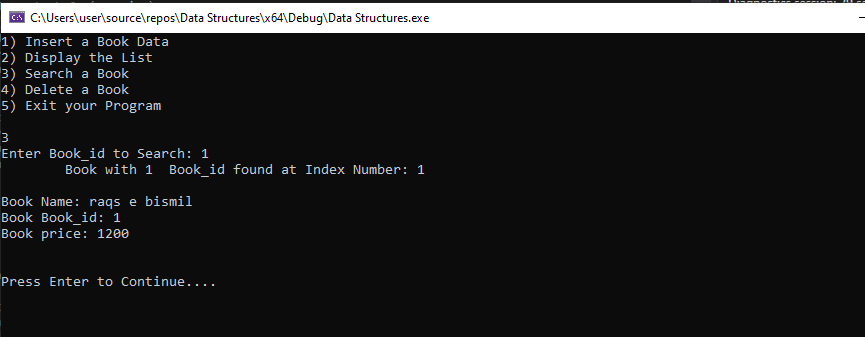
<< "\t\t\tBook price: " << arr[i].price;

}

cout << endl << endl;

}

}



**Quadratic Probing**

#include <iostream>

#include <string>

#include <windows.h>

using namespace std;

struct node

{

string name;

float Book\_id;

string quantity;

};

node arr[20];

void initialize();

int insert(node, int, int);

int hash\_fun(int, int);

void display(int);

void search(int);

void delete\_(int);

int main()

{

initialize();

int choose, Size;

while (1)

{

system("cls");

cout << "1) Insert a Book\n2) Display the List \n3) Search a Book\n4) Delete a Book\n" << endl << endl;

cin >> choose;

cin.ignore();

node temp;

switch (choose)

{

case 1:

cout << "Enter Book Name: ";

getline(cin, temp.name);

cout << "Enter Book\_id: ";

cin >> temp.Book\_id;

while (temp.Book\_id < 0 || temp.Book\_id > 100)

{

cout << "Invalid Book\_id entered.. Enter again: ";

cin >> temp.Book\_id;

}

cin.ignore();

cout << "Enter Book quantity: ";

getline(cin, temp.quantity);

Size = insert(temp, 0, 0);

if (Size == 20)

cout << "List is Full. New Data can't be Inserted...\n";

else

cout << "\nData has been Inserted...\n";

cout << endl << endl;

break;

case 2:

display(-1);

cout << "\n";

break;

case 3:

cout << "Enter Book\_id to Search: ";

cin >> choose;

cin.ignore();

search(choose);

break;

case 4:

cout << "Enter Book\_id to Delete: ";

cin >> choose;

cin.ignore();

delete\_(choose);

break;

}

cout << endl << endl << "Press Enter to Continue....";

getchar();

}

return 0;

}

void initialize()

{

for (int i = 0; i < 20; i++)

arr[i].Book\_id = -1;

}

int insert(node num, int f, int Size)

{

if (Size < 10)

{

Size++;

int index = hash\_fun(num.Book\_id, (f \* f));

if (arr[index].Book\_id == -1)

{

arr[index] = num;

}

else

return insert(num, ++f, Size);

}

return Size;

}

int hash\_fun(int num, int f)

{

return (num + f) % 10;

}

void search(int num)

{

int i = 0;

int index = hash\_fun(num, i++);

while ((arr[index].Book\_id != num) && (arr[index].Book\_id != -1) && (i < 20))

index = hash\_fun(num, i++);

if (arr[index].Book\_id == -1 || i > 20)

cout << "Book not found in the List\n";

else

{

cout << "\tBook with " << num << " Book\_id found at Index Number: " << index << "\n";

cout << "\nBook Name: " << arr[index].name << endl

<< "Book Book\_id: " << arr[index].Book\_id << endl

<< "Book quantity: " << arr[index].quantity << endl;

}

}

void delete\_(int num)

{

int i = 0;

int index = hash\_fun(num, i++);

while ((arr[index].Book\_id != num) && (arr[index].Book\_id != -1) && (i < 20))

index = hash\_fun(num, i++);

if (arr[index].Book\_id == -1 || i > 20)

cout << "Id not found in the list\n";

else

{

int zzz = arr[index].Book\_id;

arr[index].Book\_id = -1;

cout << "\tBook with " << num << " Book\_id has been deleted" << "\n";

cout << "\nBook Name: " << arr[index].name << endl

<< " Book\_id: " << zzz << endl

<< "Student quantity: " << arr[index].quantity << endl;

}

}

void display(int index = -1)

{

cout << "Book Database: \n\n";

for (int i = 0; i < 20; i++)

{

cout << "Index: " << i << "\t\t";

if (arr[i].Book\_id == -1)

cout << arr[i].Book\_id;

else

{

cout << "Book Name: " << arr[i].name << endl

<< "\t Book\_id: " << arr[i].Book\_id << endl

<< "\t\t\tBook quantity: " << arr[i].quantity;

}

cout << endl << endl;

}

}

