***TASK-01:***

#include<iostream>

#include<fstream>

#include<string>

#include<iomanip>

using namespace std;

struct Person

{

string name, location;

uint64\_t phone;

Person\* next;

};

class Contact\_Dairy

{

private:

Person\*\* contact\_list;

int list\_size;

int getCharCode(string str)

{

return str[0] - 97;

}

int HashFunc(string name)

{

int index = getCharCode(toLower(name));

return index % list\_size;

}

string toLower(string str)

{

for (int i = 0; str[i] != '\0'; i++)

{

if (str[i] >= 'A' && str[i] <= 'Z')

str[i] = str[i] + 32;

}

return str;

}

public:

Contact\_Dairy(const int size = 26)

{

list\_size = size;

contact\_list = new Person \* [list\_size];

for (int i = 0; i < list\_size; i++)

contact\_list[i] = nullptr;

}

void load\_contact()

{

ifstream fin;

string name = "", location = "";

uint64\_t phone = 0;

fin.open("Sample\_Hashing\_Data.txt");

if (!fin.is\_open())

{

cout << "File not Found!" << endl;

exit(0);

}

while (fin)

{

getline(fin, name);

fin >> phone; fin.clear(); fin.ignore();

getline(fin, location);

insert\_contact(name, phone, location);

}

fin.close();

}

void insert\_contact(string name, uint64\_t phone, string location)

{

int array\_index = HashFunc(name);

Person\* new\_contact = new Person;

new\_contact->name = name;

new\_contact->phone = phone;

new\_contact->location = location;

new\_contact->next = nullptr;

if (contact\_list[array\_index] == nullptr)

{

contact\_list[array\_index] = new\_contact;

}

else

{

Person\* root = contact\_list[array\_index];

while (root->next != nullptr)

{

root = root->next;

}

root->next = new\_contact;

}

}

bool search\_contact(string name)

{

int array\_index = HashFunc(name);

if (contact\_list[array\_index] == nullptr)

return 0;

else if (contact\_list[array\_index]->name == name)

{

cout << "Contact Found in Contact List" << endl;

cout << "Name : " << contact\_list[array\_index]->name << endl;

cout << "Phone no. : " << contact\_list[array\_index]->phone << endl;

cout << "Location : " << contact\_list[array\_index]->location << endl;

return 1;

}

else

{

Person\* current = contact\_list[array\_index];

while (current != nullptr)

{

if (current->name == name)

{

cout << "Contact Found in Contact List" << endl;

cout << "Name : " << current->name << endl;

cout << "Phone no. : " << current->phone << endl;

cout << "Location : " << current->location << endl;

return 1;

}

current = current->next;

}

}

}

bool Delete\_contact(string name)

{

int array\_index = HashFunc(name);

if (contact\_list[array\_index]->name == name)

{

contact\_list[array\_index]->name = "";

contact\_list[array\_index]->phone = 0;

contact\_list[array\_index]->location = "";

return true;

}

else

{

Person\* root = contact\_list[array\_index];

while (root != nullptr)

{

if (root->next != nullptr && root->next->name == name)

{

Person\* curr = root->next;

root->next = root->next->next;

delete curr;

return true;

}

root = root->next;

}

}

return false;

}

void display()

{

cout << "\n" << setw(55) << "Contact Dairy" << endl << endl;

for (int i = 0; i < list\_size; i++)

{

Person\* root = contact\_list[i];

if (root == nullptr)

continue;

while (root != nullptr)

{

cout << "Name : " << root->name << setw(25)

<< "Phone no. : " << root->phone << setw(25)

<< "Location : " << root->location << endl;

root = root->next;

}

}

}

};

int main()

{

int choice = 0;

string name = "", location = "";

uint64\_t phone = 0;

Contact\_Dairy lst;

lst.load\_contact();

menu:

system("cls");

cout << "\n\t1. Insert Contact into Dairy";

cout << "\n\t2. Search Contact into Dairy";

cout << "\n\t3. Delete Contact into Dairy";

cout << "\n\t4. Display";

cout << "\n\t5. Exit";

cout << "\n\tEnter Choice => ";

try

{

cin >> choice;

cin.clear(); cin.ignore();

switch (choice)

{

case 1:

cout << "Enter Contact Name : ";

getline(cin, name);

cout << "Enter Location : ";

cin >> phone; cin.clear(); cin.ignore();

cout << "Enter Contact Phone no. : ";

getline(cin, location);

if (!cin) { throw"Please Enter Correct Input"; }

else if (phone < 0) { throw"Please Enter Correct IPhone no."; }

lst.insert\_contact(name, phone, location);

break;

case 2:

cout << "Enter Contact Name : ";

getline(cin, name);

if (!lst.search\_contact(name))

cout << "Contact not found!" << endl;

break;

case 3:

cout << "Enter Contact Name : ";

getline(cin, name);

if (lst.Delete\_contact(name))

cout << "Contact Deleted Succefully!" << endl;

else

cout << "Contact not found!" << endl;

break;

case 4:

lst.display();

break;

case 5:

return 0;

break;

default:

if (!cin) { throw"Please Enter Correct Input"; }

else if (choice < 1 || choice > 5)

{

throw"Please Enter Correct Choice";

}

break;

}

system("pause");

goto menu;

}

catch (const char\* error\_msg)

{

cout << "\n\t" << error\_msg << endl;

cin.clear(); cin.ignore(); system("pause");

goto menu;

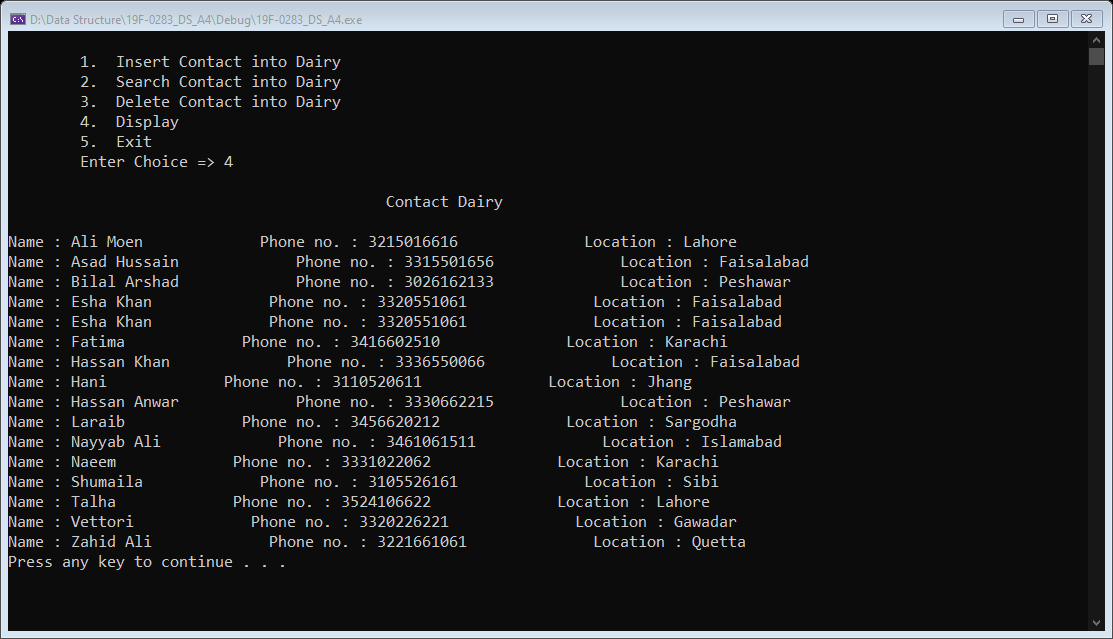
}

system("pause");

return 0;

}

***Output*** *:*

****

***TASK-02 (a):***

#include<iostream>

#include<string>

#include<iomanip>

using namespace std;

struct book

{

string name;

int quantity;

float price;

};

class book\_management

{

private:

book\* book\_list;

int list\_size;

int getCharCode(string str)

{

return str[0] - 97;

}

int HashFunc(string name)

{

int index = getCharCode(toLower(name));

return index % list\_size;

}

string toLower(string str)

{

for (int i = 0; str[i] != '\0'; i++)

{

if (str[i] >= 'A' && str[i] <= 'Z')

str[i] = str[i] + 32;

}

return str;

}

public:

book\_management(const int size = 50)

{

list\_size = size;

book\_list = new book[list\_size];

for (int i = 0; i < list\_size; i++)

{

book\_list[i].name = "";

book\_list[i].price = 0;

book\_list[i].quantity = 0;

}

}

void insert\_book(string name, int quantity, float price)

{

int array\_index = HashFunc(name);

if (book\_list[array\_index].name == "")

{

book\_list[array\_index].name = name;

book\_list[array\_index].price = price;

book\_list[array\_index].quantity = quantity;

cout << "Data of Book is inserted in Book Management" << endl;

return;

}

else

{

for (int j = 0; j < list\_size; j++)

{

array\_index = ((name[0] - 97) + j) % list\_size;

if (book\_list[array\_index].name == "")

{

book\_list[array\_index].name = name;

book\_list[array\_index].price = price;

book\_list[array\_index].quantity = quantity;

cout << "Data of Book is inserted in Book Management" << endl;

return;

}

}

}

cout << "Hash Table is Full" << endl;

}

bool search\_book(string name)

{

int array\_index = HashFunc(name);

if (book\_list[array\_index].name == name)

{

cout << "Name : " << book\_list[array\_index].name << setw(25)

<< "quantity : " << book\_list[array\_index].quantity << setw(25)

<< "price : " << book\_list[array\_index].price << endl;

return 1;

}

else

{

for (int j = 0; j < list\_size; j++)

{

array\_index = ((name[0] - 97) + j) % list\_size;

if (book\_list[array\_index].name == name)

{

cout << "Name : " << book\_list[array\_index].name << setw(25)

<< "quantity : " << book\_list[array\_index].quantity << setw(25)

<< "price : " << book\_list[array\_index].price << endl;

return 1;

break;

}

}

}

return 0;

}

bool Delete\_book(string name)

{

int array\_index = HashFunc(name);

if (book\_list[array\_index].name == name)

{

book\_list[array\_index].name = "";

book\_list[array\_index].price = 0;

book\_list[array\_index].quantity = 0;

return 1;

}

else

{

for (int j = 0; j < list\_size; j++)

{

array\_index = ((name[0] - 97) + j) % list\_size;

if (book\_list[array\_index].name == name)

{

book\_list[array\_index].name = "";

book\_list[array\_index].price = 0;

book\_list[array\_index].quantity = 0;

return 1;

break;

}

}

}

return 0;

}

void display()

{

cout << "\n" << setw(45) << "Book Management Shop" << endl << endl;

for (int i = 0; i < list\_size; i++)

{

if (book\_list[i].name != "")

{

cout << "Name : " << book\_list[i].name << setw(25)

<< "quantity : " << book\_list[i].quantity << setw(25)

<< "price : " << book\_list[i].price << endl;

}

}

}

};

int main()

{

int choice = 0;

string name = "";

float price = 0;

int quantity = 0;

book\_management lst;

menu:

system("cls");

cout << "\n\t1. Insert Book Data";

cout << "\n\t2. Search Book Data";

cout << "\n\t3. Delete Book Data";

cout << "\n\t4. Display";

cout << "\n\t5. Exit";

cout << "\n\tEnter Choice => ";

try

{

cin >> choice;

cin.clear(); cin.ignore();

switch (choice)

{

case 1:

cout << "Enter Book Name : ";

getline(cin, name);

cout << "Enter Book Price : ";

cin >> price;

if (!cin) { throw"Please Enter Correct Input"; }

else if (price < 0) { throw"Please Enter Correct Price"; }

cout << "Enter Book quantity no. : ";

cin >> quantity;

if (!cin) { throw"Please Enter Correct Input"; }

else if (quantity < 0) { throw"Please Enter Correct Quantity"; }

lst.insert\_book(name, quantity, price);

break;

case 2:

cout << "Enter Book Name : ";

getline(cin, name);

if (!lst.search\_book(name))

cout << "Book not found!" << endl;

break;

case 3:

cout << "Enter book Name : ";

getline(cin, name);

if (lst.Delete\_book(name))

cout << "Book data Deleted Succefully!" << endl;

else

cout << "Book data not found!" << endl;

break;

case 4:

lst.display();

break;

case 5:

return 0;

break;

default:

if (!cin) { throw"Please Enter Correct Input"; }

else if (choice < 1 || choice > 5)

{

throw"Please Enter Correct Choice";

}

break;

}

system("pause");

goto menu;

}

catch (const char\* error\_msg)

{

cout << "\n\t" << error\_msg << endl;

cin.clear(); cin.ignore(); system("pause");

goto menu;

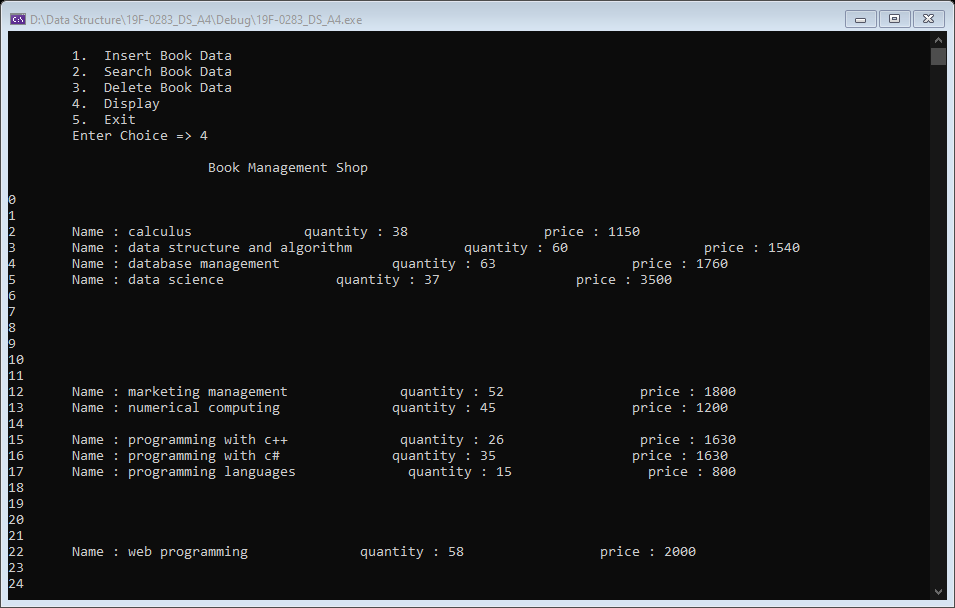
}

system("pause");

return 0;

}

**Output:**

****

***TASK-02 (b):***

#include<iostream>

#include<string>

#include<iomanip>

using namespace std;

struct book

{

string name;

int quantity;

float price;

};

class book\_management

{

private:

book\* book\_list;

int list\_size;

int getCharCode(string str)

{

return str[0] - 97;

}

int HashFunc(string name)

{

int index = getCharCode(toLower(name));

return index % list\_size;

}

string toLower(string str)

{

for (int i = 0; str[i] != '\0'; i++)

{

if (str[i] >= 'A' && str[i] <= 'Z')

str[i] = str[i] + 32;

}

return str;

}

public:

book\_management(const int size = 50)

{

list\_size = size;

book\_list = new book[list\_size];

for (int i = 0; i < list\_size; i++)

{

book\_list[i].name = "";

book\_list[i].price = 0;

book\_list[i].quantity = 0;

}

}

void insert\_book(string name, int quantity, float price)

{

int array\_index = HashFunc(name);

if (book\_list[array\_index].name == "")

{

book\_list[array\_index].name = name;

book\_list[array\_index].price = price;

book\_list[array\_index].quantity = quantity;

cout << "Data of Book is inserted in Book Management" << endl;

return;

}

else

{

for (int j = 0; j < list\_size; j++)

{

array\_index = ((name[0] - 97) + (j \* j)) % list\_size;

if (book\_list[array\_index].name == "")

{

book\_list[array\_index].name = name;

book\_list[array\_index].price = price;

book\_list[array\_index].quantity = quantity;

cout << "Data of Book is inserted in Book Management" << endl;

return;

}

}

}

cout << "Hash Table is Full" << endl;

}

bool search\_book(string name)

{

int array\_index = HashFunc(name);

if (book\_list[array\_index].name == name)

{

cout << "Name : " << book\_list[array\_index].name << setw(25)

<< "quantity : " << book\_list[array\_index].quantity << setw(25)

<< "price : " << book\_list[array\_index].price << endl;

return 1;

}

else

{

for (int j = 0; j < list\_size; j++)

{

array\_index = ((name[0] - 97) + (j \* j)) % list\_size;

if (book\_list[array\_index].name == name)

{

cout << "Name : " << book\_list[array\_index].name << setw(25)

<< "quantity : " << book\_list[array\_index].quantity << setw(25)

<< "price : " << book\_list[array\_index].price << endl;

return 1;

break;

}

}

}

return 0;

}

bool Delete\_book(string name)

{

int array\_index = HashFunc(name);

if (book\_list[array\_index].name == name)

{

book\_list[array\_index].name = "";

book\_list[array\_index].price = 0;

book\_list[array\_index].quantity = 0;

return 1;

}

else

{

for (int j = 0; j < list\_size; j++)

{

array\_index = ((name[0] - 97) + (j \* j)) % list\_size;

if (book\_list[array\_index].name == name)

{

book\_list[array\_index].name = "";

book\_list[array\_index].price = 0;

book\_list[array\_index].quantity = 0;

return 1;

break;

}

}

}

return 0;

}

void display()

{

cout << "\n" << setw(45) << "Book Management Shop" << endl << endl;

for (int i = 0; i < list\_size; i++)

{

if (book\_list[i].name != "")

{

cout << "Name : " << book\_list[i].name << setw(25)

<< "quantity : " << book\_list[i].quantity << setw(25)

<< "price : " << book\_list[i].price << endl;

}

}

}

};

int main()

{

int choice = 0;

string name = "";

float price = 0;

int quantity = 0;

book\_management lst;

menu:

system("cls");

cout << "\n\t1. Insert Book Data";

cout << "\n\t2. Search Book Data";

cout << "\n\t3. Delete Book Data";

cout << "\n\t4. Display";

cout << "\n\t5. Exit";

cout << "\n\tEnter Choice => ";

try

{

cin >> choice;

cin.clear(); cin.ignore();

switch (choice)

{

case 1:

cout << "Enter Book Name : ";

getline(cin, name);

cout << "Enter Book Price : ";

cin >> price;

if (!cin) { throw"Please Enter Correct Input"; }

else if (price < 0) { throw"Please Enter Correct Price"; }

cout << "Enter Book quantity no. : ";

cin >> quantity;

if (!cin) { throw"Please Enter Correct Input"; }

else if (quantity < 0) { throw"Please Enter Correct Quantity"; }

lst.insert\_book(name, quantity, price);

break;

case 2:

cout << "Enter Book Name : ";

getline(cin, name);

if (!lst.search\_book(name))

cout << "Book not found!" << endl;

break;

case 3:

cout << "Enter book Name : ";

getline(cin, name);

if (lst.Delete\_book(name))

cout << "Book data Deleted Succefully!" << endl;

else

cout << "Book data not found!" << endl;

break;

case 4:

lst.display();

break;

case 5:

return 0;

break;

default:

if (!cin) { throw"Please Enter Correct Input"; }

else if (choice < 1 || choice > 5)

{

throw"Please Enter Correct Choice";

}

break;

}

system("pause");

goto menu;

}

catch (const char\* error\_msg)

{

cout << "\n\t" << error\_msg << endl;

cin.clear(); cin.ignore(); system("pause");

goto menu;

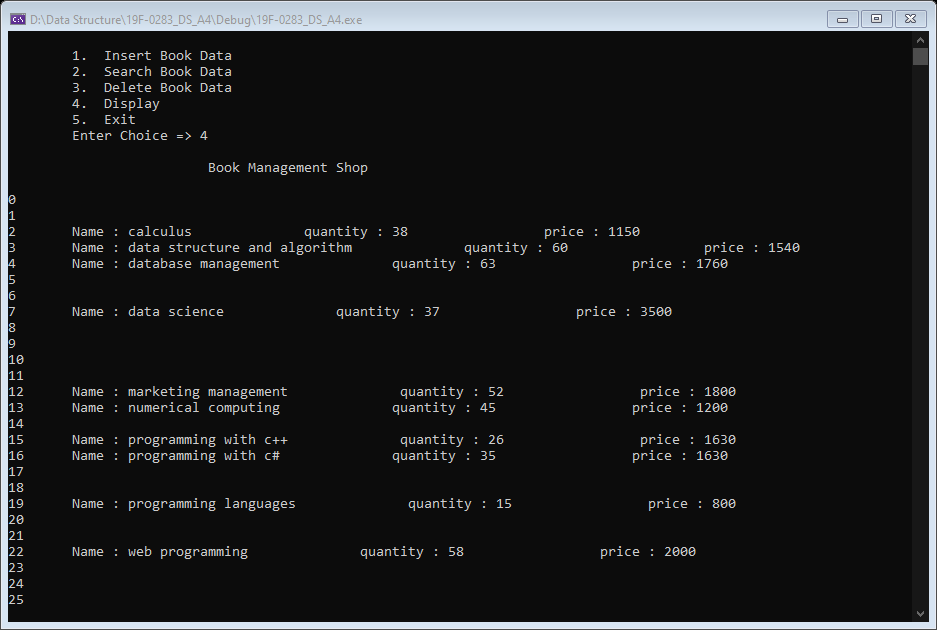
}

system("pause");

return 0;

}

**Output:**

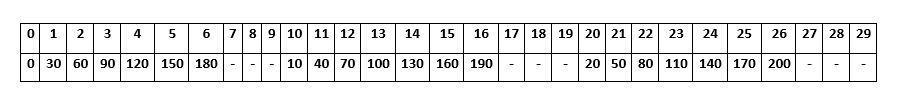


**Consider the following data values,**

**0 10 20 30 40 50 60 70 80 90 100 110 120 130 140 150 160 170 180 190 200**

**Your task is to take an array of size 30 and insert these values into the array using**

1. **Linear Probing**

****

**0 % 30 = 0**

**10 % 30 = 10**

**20 % 30 = 20**

**30 + 0 % 30 = 0**

**30 + 1 % 30 = 1**

**40 + 0 % 30 = 10**

**40 + 1 % 30 = 11**

**50 + 0 % 30 = 20**

**50 + 1 % 30 = 21**

**60 + 0 % 30 = 0**

**60 + 1 % 30 = 1**

**60 + 2 % 30 = 2**

**70 + 0 % 30 = 10**

**70 + 1 % 30 = 11**

**70 + 2 % 30 = 12**

**80 + 0 % 30 = 20**

**80 + 1 % 30 = 21**

**80 + 2 % 30 = 22**

**90 + 0 % 30 = 0**

**90 + 1 % 30 = 1**

**90 + 2 % 30 = 2**

**90 + 3 % 30 = 3**

**100 + 0 % 30 = 10**

**100 + 1 % 30 = 11**

**100 + 2 % 30 = 12**

**100 + 3 % 30 = 13**

**110 + 0 % 30 = 20**

**110 + 1 % 30 = 21**

**110 + 2 % 30 = 22**

**110 + 3 % 30 = 23**

**120 + 0 % 30 = 0**

**120 + 1 % 30 = 1**

**120 + 2 % 30 = 2**

**120 + 3 % 30 = 3**

**120 + 4 % 30 = 4**

**130 + 0 % 30 = 10**

**130 + 1 % 30 = 11**

**130 + 2 % 30 = 12**

**130 + 3 % 30 = 13**

**130 + 4 % 30 = 14**

**140 + 0 % 30 = 20**

**140 + 1 % 30 = 21**

**140 + 2 % 30 = 22**

**140 + 3 % 30 = 23**

**140 + 4 % 30 = 24**

**150 + 0 % 30 = 0**

**150 + 1 % 30 = 1**

**150 + 2 % 30 = 2**

**150 + 3 % 30 = 3**

**150 + 4 % 30 = 4**

**150 + 5 % 30 = 5**

**160 + 0 % 30 = 10**

**160 + 1 % 30 = 11**

**160 + 2 % 30 = 12**

**160 + 3 % 30 = 13**

**160 + 4 % 30 = 14**

**160 + 5 % 30 = 15**

**170 + 0 % 30 = 20**

**170 + 1 % 30 = 21**

**170 + 2 % 30 = 22**

**170 + 3 % 30 = 23**

**170 + 4 % 30 = 24**

**170 + 5 % 30 = 25**

**180 + 0 % 30 = 0**

**180 + 1 % 30 = 1**

**180 + 2 % 30 = 2**

**180 + 3 % 30 = 3**

**180 + 4 % 30 = 4**

**180 + 5 % 30 = 5**

**180 + 6 % 30 = 6**

**190 + 0 % 30 = 10**

**190 + 1 % 30 = 11**

**190 + 2 % 30 = 12**

**190 + 3 % 30 = 13**

**190 + 4 % 30 = 14**

**190 + 5 % 30 = 15**

**190 + 6 % 30 = 16**

**200 + 0 % 30 = 20**

**200 + 1 % 30 = 21**

**200 + 2 % 30 = 22**

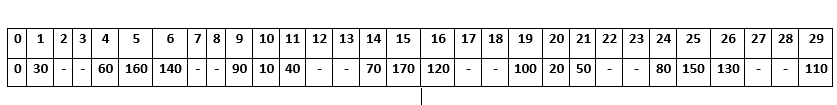
**200 + 3 % 30 = 23**

**200 + 4 % 30 = 24**

**200 + 5 % 30 = 25**

**200 + 6 % 30 = 26**

1. **Quadratic Probing**

****

**0 % 30 = 0**

**10 % 30 = 10**

**20 % 30 = 20**

**30 + 0 % 30 = 0**

**30 + 1 % 30 = 1**

**40 + 0 % 30 = 10**

**40 + 1 % 30 = 11**

**50 + 0 % 30 = 20**

**50 + 1 % 30 = 21**

**60 + 0 % 30 = 0**

**60 + 1 % 30 = 1**

**60 + 4 % 30 = 4**

**70 + 0 % 30 = 10**

**70 + 1 % 30 = 11**

**70 + 4 % 30 = 14**

**80 + 0 % 30 = 20**

**80 + 1 % 30 = 21**

**80 + 4 % 30 = 24**

**90 + 0 % 30 = 0**

**90 + 1 % 30 = 1**

**90 + 4 % 30 = 4**

**90 + 9 % 30 = 9**

**100 + 0 % 30 = 10**

**100 + 1 % 30 = 11**

**100 + 4 % 30 = 14**

**100 + 9 % 30 = 19**

**110 + 0 % 30 = 20**

**110 + 1 % 30 = 21**

**110 + 4 % 30 = 24**

**110 + 9 % 30 = 29**

**120 + 0 % 30 = 0**

**120 + 1 % 30 = 1**

**120 + 4 % 30 = 4**

**120 + 9 % 30 = 9**

**120 + 16 % 30 = 16**

**130 + 0 % 30 = 10**

**130 + 1 % 30 = 11**

**130 + 4 % 30 = 14**

**130 + 9 % 30 = 19**

**130 + 16 % 30 = 26**

**140 + 0 % 30 = 20**

**140 + 1 % 30 = 21**

**140 + 4 % 30 = 24**

**140 + 9 % 30 = 29**

**140 + 16 % 30 = 6**

**150 + 0 % 30 = 0**

**150 + 1 % 30 = 1**

**150 + 4 % 30 = 4**

**150 + 9 % 30 = 9**

**150 + 16 % 30 = 16**

**150 + 25 % 30 = 25**

**160 + 0 % 30 = 10**

**160 + 1 % 30 = 11**

**160 + 4 % 30 = 14**

**160 + 9 % 30 = 19**

**160 + 16 % 30 = 26**

**160 + 25 % 30 = 5**

**170 + 0 % 30 = 20**

**170 + 1 % 30 = 21**

**170 + 4 % 30 = 24**

**170 + 9 % 30 = 29**

**170 + 16 % 30 = 6**

**170 + 25 % 30 = 15**

**180 + 0 % 30 = 0**

**180 + 1 % 30 = 1**

**180 + 4 % 30 = 4**

**180 + 9 % 30 = 9**

**180 + 16 % 30 = 16**

**180 + 25 % 30 = 25**

**180 + 36 % 30 = 6**

**180 + 49 % 30 = 19**

**180 + 64 % 30 = 4**

**180 + 81 % 30 = 21**

**180 + 100 % 30 = 10**

**180 + 121 % 30 = 1**

**180 + 144 % 30 = 24**

**180 + 169 % 30 = 19**

**180 + 196 % 30 = 16**

**180 + 225 % 30 = 15**

**180 + 256 % 30 = 16**

**180 + 289 % 30 = 19**

**180 + 324 % 30 = 24**

**180 + 361 % 30 = 1**

**180 + 400 % 30 = 10**

**180 + 441 % 30 = 21**

**180 + 484 % 30 = 4**

**180 + 529 % 30 = 19**

**180 + 576 % 30 = 6**

**180 + 625 % 30 = 25**

**180 + 676 % 30 = 16**

**180 + 729 % 30 = 9**

**180 + 784 % 30 = 4**

**180 + 841 % 30 = 1**

**180 =Index not found**

**190 + 0 % 30 = 10**

**190 + 1 % 30 = 11**

**190 + 4 % 30 = 14**

**190 + 9 % 30 = 19**

**190 + 16 % 30 = 26**

**190 + 25 % 30 = 5**

**190 + 36 % 30 = 16**

**190 + 49 % 30 = 29**

**190 + 64 % 30 = 14**

**190 + 81 % 30 = 1**

**190 + 100 % 30 = 20**

**190 + 121 % 30 = 11**

**190 + 144 % 30 = 4**

**190 + 169 % 30 = 29**

**190 + 196 % 30 = 26**

**190 + 225 % 30 = 25**

**190 + 256 % 30 = 26**

**190 + 289 % 30 = 29**

**190 + 324 % 30 = 4**

**190 + 361 % 30 = 11**

**190 + 400 % 30 = 20**

**190 + 441 % 30 = 1**

**190 + 484 % 30 = 14**

**190 + 529 % 30 = 29**

**190 + 576 % 30 = 16**

**190 + 625 % 30 = 5**

**190 + 676 % 30 = 26**

**190 + 729 % 30 = 19**

**190 + 784 % 30 = 14**

**190 + 841 % 30 = 11**

**190=Index not found**

**200 + 0 % 30 = 20**

**200 + 1 % 30 = 21**

**200 + 4 % 30 = 24**

**200 + 9 % 30 = 29**

**200 + 16 % 30 = 6**

**200 + 25 % 30 = 15**

**200 + 36 % 30 = 26**

**200 + 49 % 30 = 9**

**200 + 64 % 30 = 24**

**200 + 81 % 30 = 11**

**200 + 100 % 30 = 0**

**200 + 121 % 30 = 21**

**200 + 144 % 30 = 14**

**200 + 169 % 30 = 9**

**200 + 196 % 30 = 6**

**200 + 225 % 30 = 5**

**200 + 256 % 30 = 6**

**200 + 289 % 30 = 9**

**200 + 324 % 30 = 14**

**200 + 361 % 30 = 21**

**200 + 400 % 30 = 0**

**200 + 441 % 30 = 11**

**200 + 484 % 30 = 24**

**200 + 529 % 30 = 9**

**200 + 576 % 30 = 26**

**200 + 625 % 30 = 15**

**200 + 676 % 30 = 6**

**200 + 729 % 30 = 29**

**200 + 784 % 30 = 24**

**200 + 841 % 30 = 21**

**200= Index not found**