#include <iostream>

using namespace std;

class customer

{

public:

int ID;

string name;

int age;

char sex;

int incomeRange;

string segment;

};

class Node

{

public:

Node \*next;

Node \*prev;

customer data;

Node(customer data)

{

this->data = data;

next = NULL;

prev = NULL;

}

};

class IDList

{

public:

Node \*head;

Node \*tail;

int count;

IDList()

{

head = NULL;

tail = NULL;

count = 0;

}

void PrintList()

{

Node \*cur = head;

while (cur != NULL)

{

cout << cur->data.ID << " " << cur->data.name << " " << cur->data.age << " " << cur->data.sex << " " << cur->data.incomeRange << " " << cur->data.segment << endl;

cur = cur->next;

}

cout << endl;

}

void addNode(customer data)

{

Node \*newNode = new Node(data);

Node \*cur = head;

if (head == NULL)

{

head = newNode;

tail = newNode;

count++;

return;

}

if (head->data.ID > data.ID)

{

newNode->next = head;

head->prev = newNode;

head = newNode;

count++;

return;

}

if (tail->data.ID < data.ID)

{

tail->next = newNode;

newNode->prev = tail;

tail = newNode;

count++;

return;

}

while (cur->next != NULL)

{

if (cur->data.ID < data.ID && cur->next->data.ID > data.ID)

{

newNode->next = cur->next;

newNode->prev = cur;

cur->next->prev = newNode;

cur->next = newNode;

count++;

return;

}

cur = cur->next;

}

}

void addNodebyIncome(customer data)

{

Node \*newNode = new Node(data);

Node \*cur = head;

if (head == NULL)

{

head = newNode;

tail = newNode;

count++;

return;

}

if (head->data.incomeRange > data.incomeRange)

{

newNode->next = head;

head->prev = newNode;

head = newNode;

count++;

return;

}

if (tail->data.incomeRange < data.incomeRange)

{

tail->next = newNode;

newNode->prev = tail;

tail = newNode;

count++;

return;

}

while (cur->next != NULL)

{

if (cur->data.incomeRange < data.incomeRange && cur->next->data.incomeRange > data.incomeRange)

{

newNode->next = cur->next;

newNode->prev = cur;

cur->next->prev = newNode;

cur->next = newNode;

count++;

return;

}

cur = cur->next;

}

}

void addbyAge(customer data)

{

Node \*newNode = new Node(data);

Node \*cur = head;

if (head == NULL)

{

head = newNode;

tail = newNode;

count++;

return;

}

if (head->data.age > data.age)

{

newNode->next = head;

head->prev = newNode;

head = newNode;

count++;

return;

}

if (tail->data.age < data.age)

{

tail->next = newNode;

newNode->prev = tail;

tail = newNode;

count++;

return;

}

while (cur->next != NULL)

{

if (cur->data.age < data.age && cur->next->data.age > data.age)

{

newNode->next = cur->next;

newNode->prev = cur;

cur->next->prev = newNode;

cur->next = newNode;

count++;

return;

}

cur = cur->next;

}

}

};

int main(){

int n;

cout << "Enter the number of customers : ";

cin >> n;

customer \*arrayList = new customer[n];

IDList \*idList = new IDList();

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

{

int ID[] = {1,3,8,2,9,4,5,6,7,10};

arrayList[i].ID = ID[i];

string name[] = {"tenten","sasuke","naruto","sakura","kakashi","minato","kushina","jiraya","orochimaru","madara"};

arrayList[i].name = name[i];

int age[] = { 20, 21, 29,27,30,22,60,50,40,35};

arrayList[i].age = age[i];

char sex[] = {'F','M','M','F','M','M','F','M','M','M'};

arrayList[i].sex = sex[i];

int incomeRange[] = {20000,50000,35000,9000,27000,90000,100100,5400,100000,55000};

arrayList[i].incomeRange = incomeRange[i];

string segment[] = {"target","non-target","non-target","target","non-target","non-target","non-target","target","non-target","non-target"};

arrayList[i].segment = segment[i];

// case 1 : sort by ID

idList->addNode(arrayList[i]);

// case 2 : sort by incomeRange

idList->addNodebyIncome(arrayList[i]);

// case 3 : sort by age

idList -> addbyAge(arrayList[i]);

}

idList->PrintList();

}

Output :

Case H1 By ID :

Text

Description automatically generated

Case H2 By IncomeRange :

Text

Description automatically generated

Case H3 By Age:

Text

Description automatically generated