64090500421
นายวัชรพล ตรีสัตยสกุล

5. กำหนดให้เรามีโครงสร้างข้อมูลอยู่หลายแอตทริบิวท์ เช่น

```
class customer{
private:
    int ID;
     string name;
     int age;
     int sex;
     int incomeRange;
     int segment;
     }
```

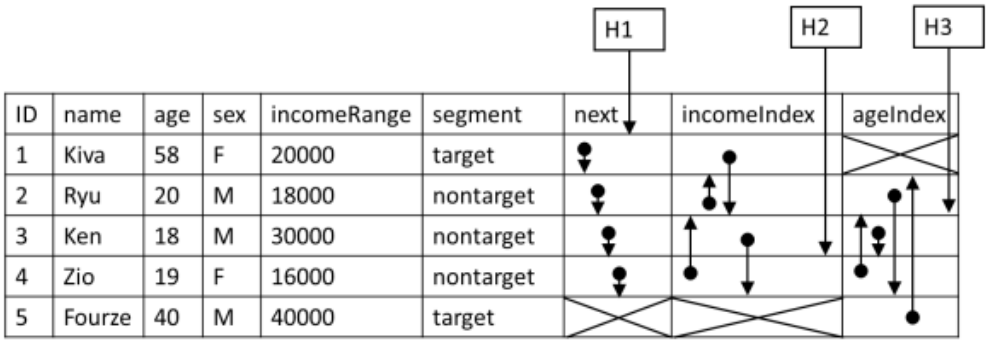
เราสามารถนำมาสร้างได้ทั้ง ArrayList ได้ดังนี้

ID	name	age	sex	incomeRange	segment
1	Kiva	58	F	20000	target
2	Ryu	20	M	18000	nontarget
3	Ken	18	M	30000	nontarget
4	Zio	19	F	16000	nontarget
5	Fourze	40	M	40000	target

เพื่อความเร็วในการเข้าถึงข้อมูลในแต่ละแอตทริบิวท์เราจึงสร้างดัชนีเพื่อใช้ในการค้นหาข้อมูลโดยเรียงลำดับจากน้อยไปหามากดังนี้

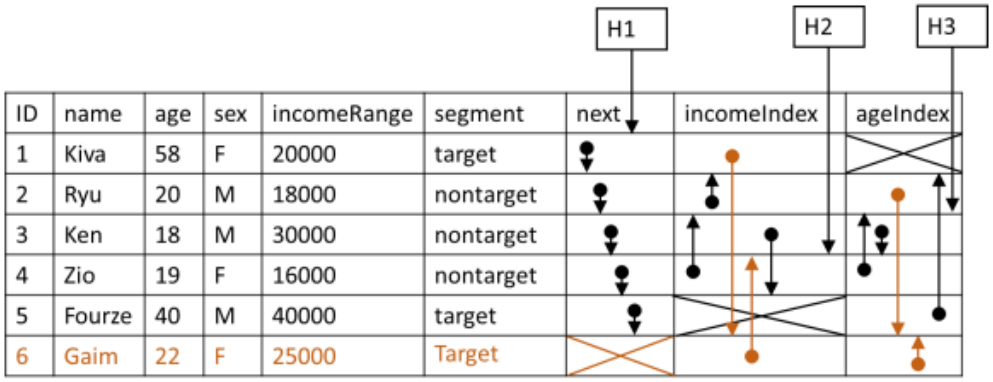
64090500421

นายวัชรพล ตรีสัตยสกุล



ID	name	age	sex	incomeRange	segment	next	incomeIndex	ageIndex
1	Kiva	58	F	20000	target			
2	Ryu	20	M	18000	nontarget			
3	Ken	18	M	30000	nontarget			
4	Zio	19	F	16000	nontarget			
5	Fourze	40	M	40000	target			

ผลของการสร้าง sort index ทำให้เราสามารถนำข้อมูลมาแสดงตามลำดับได้เลย $O(n)$ โดยไม่ต้องเรียงใหม่ แต่ไม่สามารถค้นหาข้อมูลตามแอตทริบิวท์ที่ต้องการในเวลาไม่เกิน $O(\log n)$ เนื่องจากไม่สามารถคำนวณค่ากลางระหว่างลิสต์ได้ โดยการนำหน่วยความจำไปแลกกับเวลา



ID	name	age	sex	incomeRange	segment	next	incomeIndex	ageIndex
1	Kiva	58	F	20000	target			
2	Ryu	20	M	18000	nontarget			
3	Ken	18	M	30000	nontarget			
4	Zio	19	F	16000	nontarget			
5	Fourze	40	M	40000	target			
6	Gaim	22	F	25000	Target			

ทุกครั้งที่เราทำการเพิ่มข้อมูลในลิสต์หลักเราก็เพิ่มข้อมูลด้วย append โดยใช้เวลา $O(1)$ เราก็ทำการจัดเรียงข้อมูลของดัชนีไปด้วยแบบ insertionSort $O(n)$ และมีกรณีเฉลี่ยคือ $O(n/2)$ แบบมองโลกในแง่ดี จงเขียนโปรแกรมเพื่อการจัดเรียงดัชนี

64090500421

นายวัชรพล ตรีสัตยสกุล

Code : https://ideone.com/SbMq4K

```
#include<iostream>
#include<string.h>
using namespace std;


class customer
{
    public:
        int ID;
        string name;
        int age;
        char sex;
        int incomeRange;
        bool segment;
        customer *next;
};


class arrange
{
    public:
        customer *head_id;
        customer *head_age;
        customer *head_income;
        customer *fence;

    public:
        arrange()
        {
            head_id = NULL;
            head_age = NULL;
            head_income = NULL;
        }

    void H1(int id, string name, int age, char sex, int income, bool segment)
    {
        customer* curr = new customer();
        curr->ID = id;
        curr->age = age;
        curr->name = name;
        curr->incomeRange = income;
        curr->segment = segment;
        curr->sex = sex;
        curr->next = NULL;

        if(head_id == NULL)
        {
            head_id = curr;
        }
        else
```

64090500421
นายวัชรพล ตรีสัตยสกุล

```
51          {
52              if(curr->ID < head_id->ID)
53              {
54                  curr->next = head_id;
55                  head_id = curr;
56              }
57              else
58              {
59                  fence = head_id;
60                  while(fence->next != NULL)
61                  {
62                      if(curr->ID < fence->next->ID)
63                      {
64                          curr->next = fence->next;
65                          fence->next = curr;
66                          return ;
67                      }
68                      fence = fence->next;
69                  }
70                  fence->next = curr;
71              }
72          }
73      }
74
75      void H2(int id, string name, int age, char sex, int income, bool segment)
76      {
77          customer* curr = new customer();
78          curr->ID = id;
79          curr->age = age;
80          curr->name = name;
81          curr->incomeRange = income;
82          curr->segment = segment;
83          curr->sex = sex;
84          curr->next = NULL;
85
86          if(head_age == NULL)
87          {
88              head_age = curr;
89          }
90          else
91          {
92              if(curr->age < head_age->age)
93              {
94                  curr->next = head_age;
95                  head_age = curr;
96              }
97              else
98              {
99                  fence = head_age;
100                 while(fence->next != NULL)
```

64090500421
นายวัชรพล ตรีสัตยสกุล

```
101                 {
102                     if(curr->age < fence->next->age)
103                     {
104                         curr->next = fence->next;
105                         fence->next = curr;
106                         return ;
107                     }
108                     fence = fence->next;
109                 }
110                 fence->next = curr;
111             }
112         }
113     }
114
115     void H3(int id, string name, int age, char sex, int income, bool segment)
116     {
117         customer* curr = new customer();
118         curr->ID = id;
119         curr->age = age;
120         curr->name = name;
121         curr->incomeRange = income;
122         curr->segment = segment;
123         curr->sex = sex;
124         curr->next = NULL;
125
126         if(head_income == NULL)
127         {
128             head_income = curr;
129         }
130         else
131         {
132             if(curr->incomeRange < head_income->incomeRange)
133             {
134                 curr->next = head_income;
135                 head_income = curr;
136             }
137             else
138             {
139                 fence = head_income;
140                 while(fence->next != NULL)
141                 {
142                     if(curr->incomeRange < fence->next->incomeRange)
143                     {
144                         curr->next = fence->next;
145                         fence->next = curr;
146                         return ;
147                     }
148                     fence = fence->next;
149                 }
150                 fence->next = curr;
```

64090500421
นายวัชรพล ตรีสัตยสกุล

```
151             }
152         }
153     }
154
155     void append(int id, string name, int age, char sex, int income, bool segment)
156     {
157         H1(id, name, age, sex, income, segment);
158         H2(id, name, age, sex, income, segment);
159         H3(id, name, age, sex, income, segment);
160     }
161
162     void show()
163     {
164         fence = head_id;
165         cout << "sort by ID ...\n";
166         for(;fence != NULL;)
167         {
168                 cout << fence->ID << "\t" << fence->name << "\t" << fence->age << "\t" <<
    fence->sex << "\t" << fence->incomeRange << endl;
169                 fence = fence->next;
170         }
171         fence = head_age;
172         cout << "sort by Age ...\n";
173         for(;fence != NULL;)
174         {
175                 cout << fence->ID << "\t" << fence->name << "\t" << fence->age << "\t" <<
    fence->sex << "\t" << fence->incomeRange << endl;
176                 fence = fence->next;
177         }
178         fence = head_income;
179         cout << "sort by IncomeRange ...\n";
180         for(;fence != NULL;)
181         {
182                 cout << fence->ID << "\t" << fence->name << "\t" << fence->age << "\t" <<
    fence->sex << "\t" << fence->incomeRange << endl;
183                 fence = fence->next;
184         }
185         cout << endl;
186     }
187 };
188
189 int main()
190 {
191     arrange data;
192     data.append(1,"Kiva",58,'F',20000,true);
193     data.append(2,"Ryu",20,'M',18000,false);
194     data.append(3,"Ken",18,'M',30000,false);
195     data.append(4,"Zio",19,'F',16000,false);
196     data.append(5,"Fourze",40,'F',40000,true);
197     data.show();
198     data.append(6,"Gaim",22,'F',25000,true);
199     data.show();
200 }
201
```

OUTPUT

```
sort by ID ...
1       Kiva    58      F       20000
2       Ryu     20      M       18000
3       Ken     18      M       30000
4       Zio     19      F       16000
5       Fourze  40      F       40000
sort by Age ...
3       Ken     18      M       30000
4       Zio     19      F       16000
2       Ryu     20      M       18000
5       Fourze  40      F       40000
1       Kiva    58      F       20000
sort by IncomeRange ...
4       Zio     19      F       16000
2       Ryu     20      M       18000
1       Kiva    58      F       20000
3       Ken     18      M       30000
5       Fourze  40      F       40000

sort by ID ...
1       Kiva    58      F       20000
2       Ryu     20      M       18000
3       Ken     18      M       30000
4       Zio     19      F       16000
5       Fourze  40      F       40000
6       Gaim    22      F       25000
sort by Age ...
3       Ken     18      M       30000
4       Zio     19      F       16000
2       Ryu     20      M       18000
6       Gaim    22      F       25000
5       Fourze  40      F       40000
1       Kiva    58      F       20000
sort by IncomeRange ...
4       Zio     19      F       16000
2       Ryu     20      M       18000
1       Kiva    58      F       20000
6       Gaim    22      F       25000
3       Ken     18      M       30000
5       Fourze  40      F       40000
```

---------- ข้อ 5 ----------