ระบบเวลาในการออกจากบ้านของแต่ละครอบครัวในช่วงปิดเมือง

จัดทำโดย

นายพสวัต แตงอ่อน รหัสนักศึกษา 6213129

นายวสวัตติ์ เพ็งประโคน รหัสนักศึกษา 6213132

นักศึกษาชั้นปีที่ 1 มหาวิทยาลัยมหิดล

เสนอ

อาจารย์ ดร.มิ่งมานัส ศิวรักษ์

รายวิชา EGCO112 Programming Techniques ภาคเรียนที่ 2 ปีการศึกษา 2562

สารบัญ

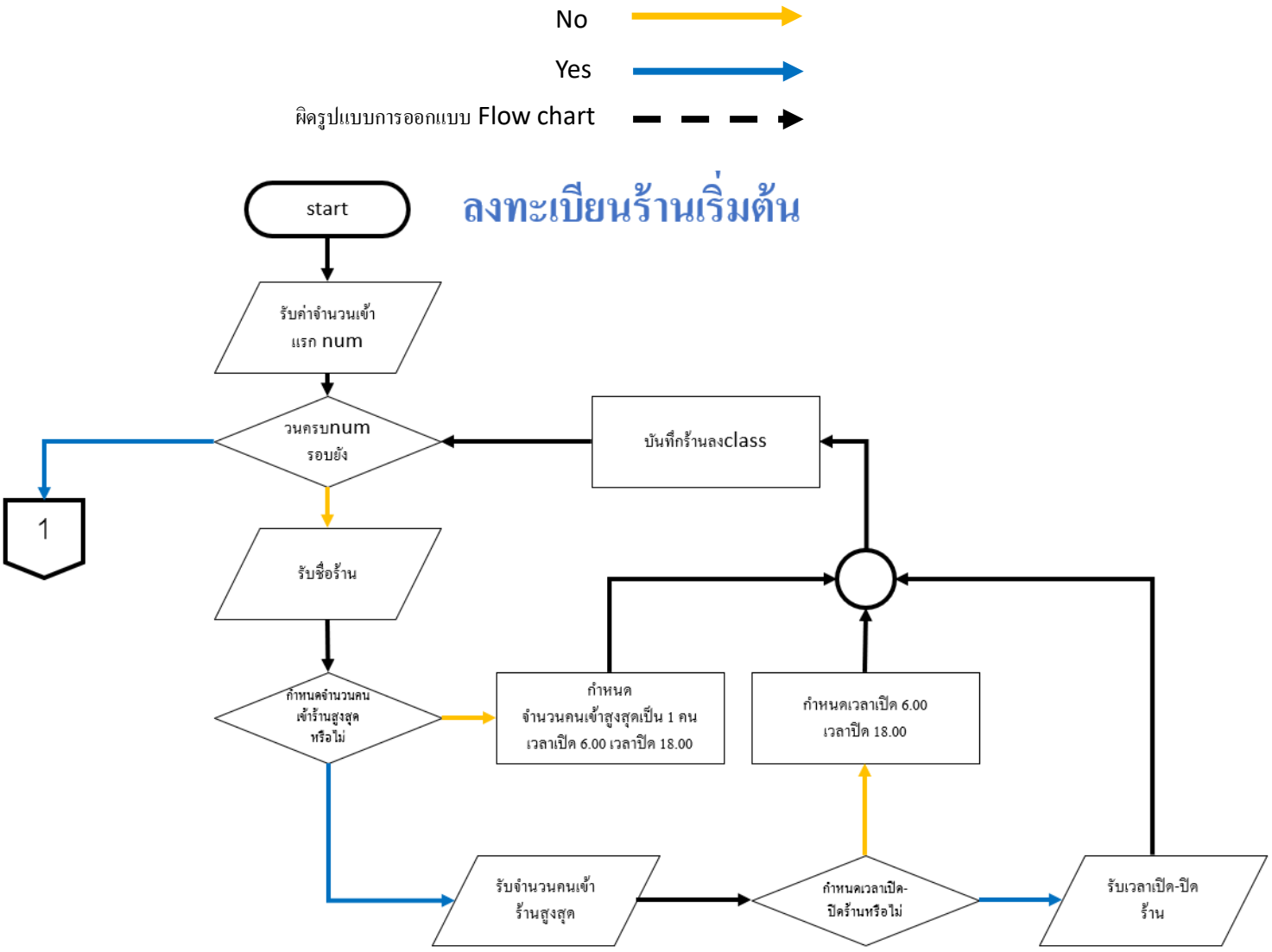
หัวข้อ	หน้า
1. แรงจูงใจในการพัฒนาโปรแกรม	1
2. การทำงานของโปรแกรม	1
2.1 Flowchart	1
2.2 ตัวอย่างการใช้งานโปรแกรม	3
3. Class template	5
3.1 Class order	5
3.2 Class building	6
3.3 Class family	7
3.4 Class town	8
4. ข้อกำหนด	8
4.1 Linked list	8
4.2 Sorting Algorithm	8
4.3 Class with constructor	9
4.4 Polymorphism	10
4.5 Exception Handling	11
5. ข้อจำกัด	12

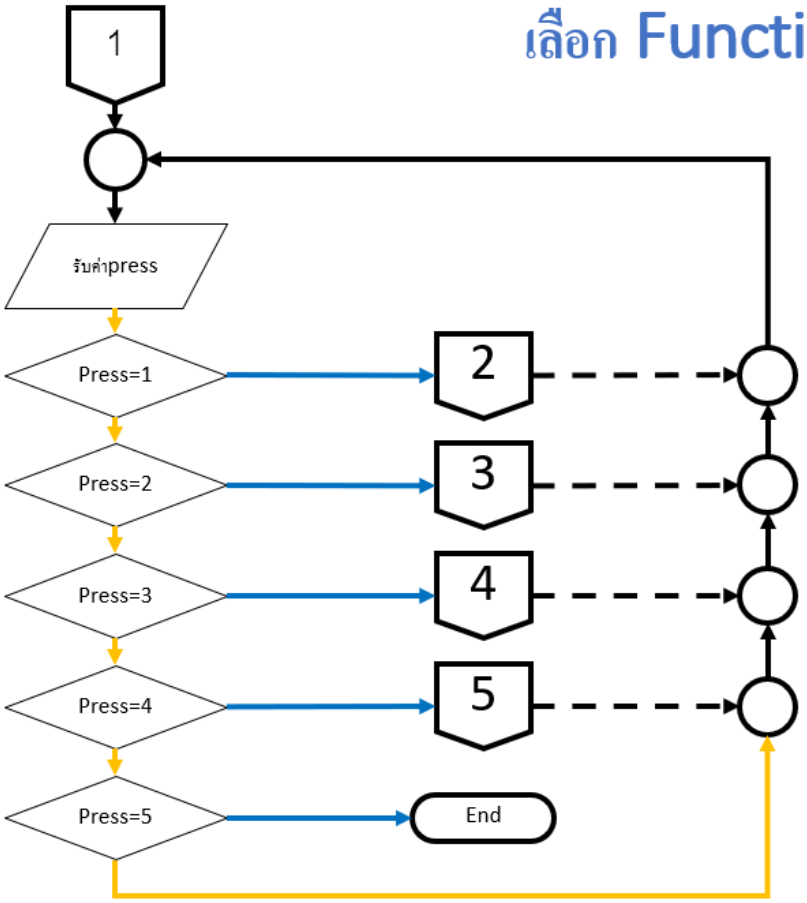
1. แรงจูงใจในการพัฒนาโปรแกรม

เนื่องจากผลการทบจากสถานการณ์การระบาดของโรค COVID-19 ทำให้หลาย ๆ จังหวัดในประเทศไทยได้ทำการประกาศปิดเมืองและประกาศเคอร์ฟิว ทำให้ผู้คนในเมืองสามารถออกจากเมืองได้ในช่วงเวลาที่กำหนดเท่านั้น จากในกรณีของเมืองอู่ฮั่น ประเทศจีน แต่ละครอบครัวสามารถออกมาซื้อของได้แค่ 2 วันต่อครั้ง และบางร้านค้านั้นก็จำกัดจำนวนผู้เข้าเพื่อรักษามาตรการเว้นระยะห่างทางสังคมทางผู้จัดทำจึงต้องการวางแผนเวลาการเดินทางไปซื้อของของแต่ละครอบครัวเพื่อให้แต่ละร้านค้าสามารถควบคุมจำนวนคนเข้าร้านและเก็บข้อมูลว่าครอบครัวไหนเข้ามาในร้านค้าได้บ้าง ซึ่งถ้าหากเกิดการติดเชื้อขึ้นมา ข้อมูลนี้จะเป็นประโยชน์ในการระบุสถานที่ที่ผู้ติดเชื้อไปในแต่ละช่วงเวลาได้

2. การทำงานของโปรแกรม

2.1 Flowchart





2 ลงทะเบียนร้านค้าเพิ่ม

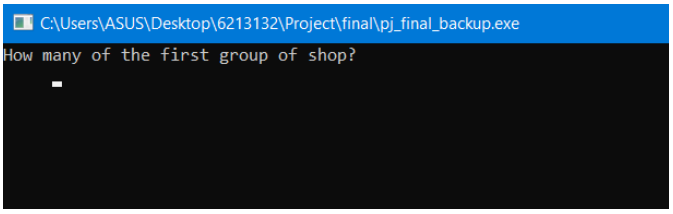
3 ลงทะเบียนครอบครัวและจองเวลา

4 ดูตารางเวลาของแต่ละร้านค้า

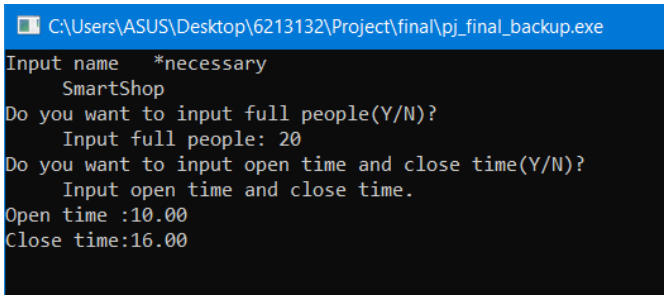
5 ดูเวลาของแต่ละครอบครัว

2.2 ตัวอย่างการใช้งานโปรแกรม

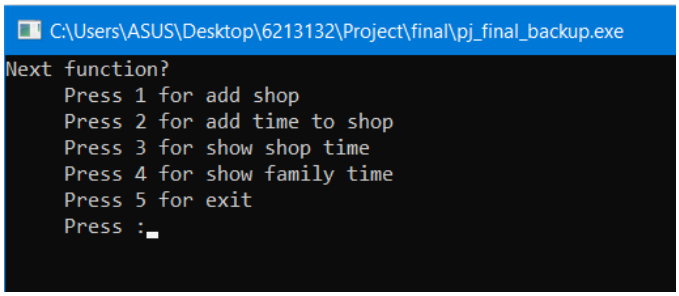
2.2.1 เมื่อเปิด Program ขึ้นมา จะให้ผู้ใช้กรอกจำนวนร้านค้าเริ่มว่ามีกี่ร้าน



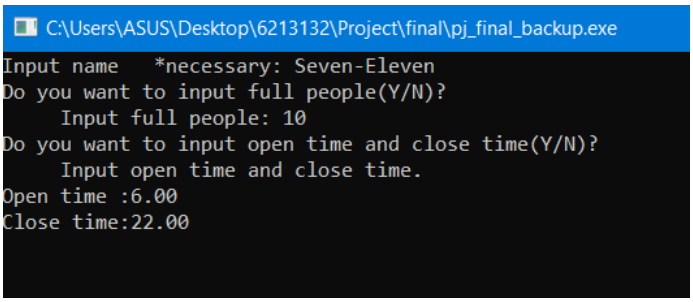
2.2.2 เมื่อกรอกตัวเลขลงไป ต่อไปโปรแกรมจะให้ผู้ใช้ใส่รายละเอียดของร้านค้า



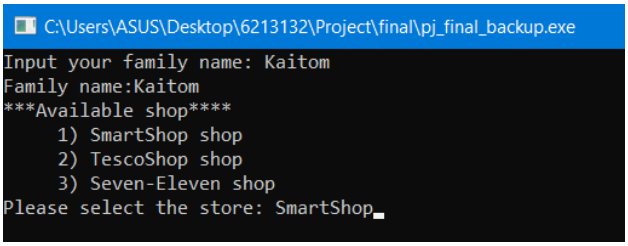
2.2.3 หลังจากกรอกข้อมูลจนครบแล้ว ต่อไปจะนำผู้ใช้เข้าสู่หน้าเมนูหลักเพื่อเลือกคำสั่งต่อไป

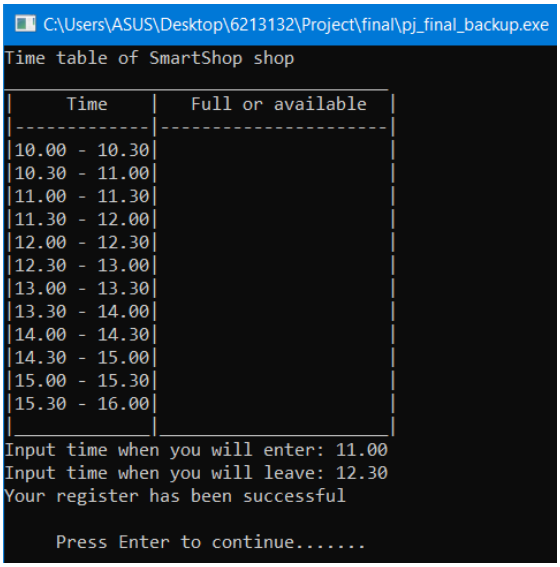


2.2.4 ที่หน้าเมนูหลัก หากกดเลข 1 จะนำผู้ใช้ไปสู่หน้าเพิ่มร้านค้า เหมือนกับ 2.2.2 และเมื่อเสร็จแล้ว จะกลับมาที่หน้าเมนูหลักเหมือนเดิม

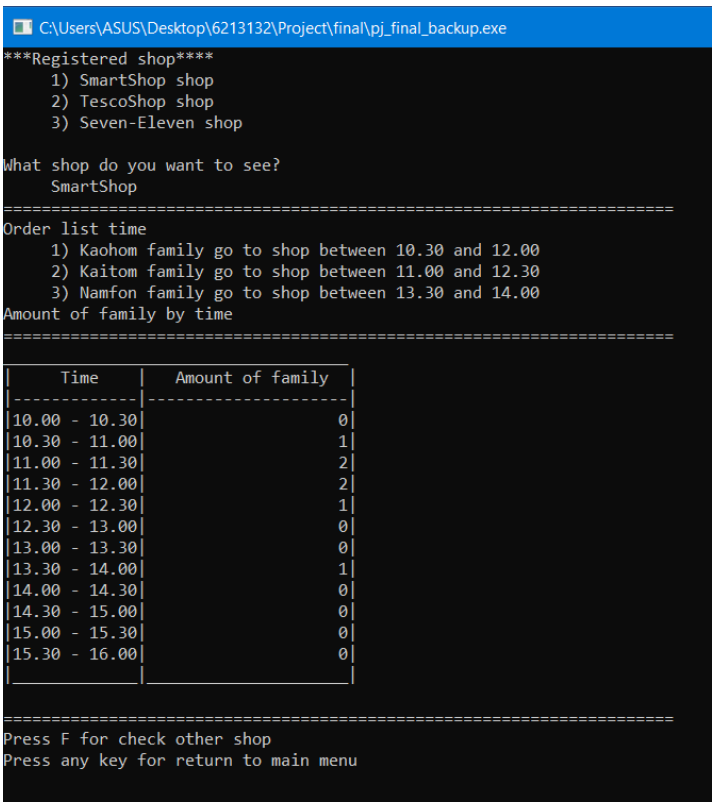


2.2.5 ที่หน้าเมนูหลัก หากกดเลข 2 จะไปสู่หน้าลงทะเบียนครอบครัวและจองเวลา โดยผู้ใช้ต้องกรอกชื่อครอบครัว ร้านค้าที่ต้องการจองเวลา จากนั้น ผู้ใช้ต้องกรอกเวลาที่ต้องการจอง หากเวลาที่จองไม่สามารถจองได้ จะขึ้นเหตุผลว่าเพราะอะไรและให้ผู้ใช้เลือกกรอกใหม่หรือกลับไปหน้าเลือกร้านค้า หากจองได้จะขึ้นประโยคยืนยันว่าจองสำเร็จ และจะให้ผู้ใช้เลือกว่าต้องการจองเวลาเพิ่มหรือมีครอบครัวอื่นต้องการลงทะเบียนอีกไหม หากไม่ จะนำผู้ใช้กลับไปหน้าเมนูหลัก

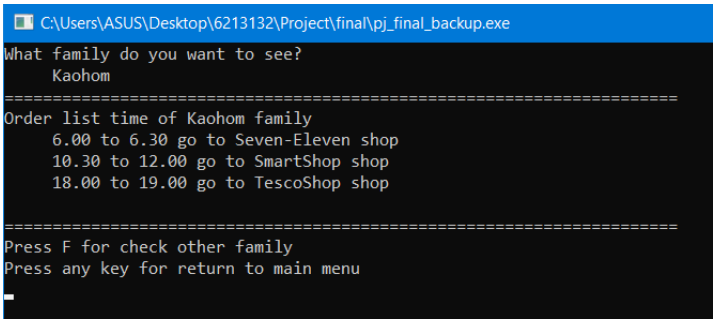




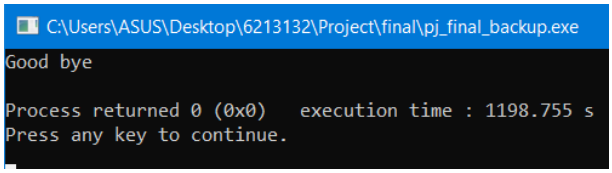
2.2.7 ที่หน้าเมนูหลัก หากกดเลข 3 จะเป็นการดูตารางเวลาของแต่ละร้านค้า และแสดงว่ามีครอบครัวไหนจองเวลาใดไปบ้าง



2.2.8 ที่หน้าเมนูหลัก หากกดเลข 4 จะเป็นการดูตารางเวลาของแต่ละครอบครัวว่าได้จองร้านค้าไหนที่เวลาใดบ้าง



2.2.9 ที่หน้าเมนูหลัก หากกดเลข 5 จะเป็นการออกจากโปรแกรมและสิ้นสุดการทำงาน



3. Class template

3.1 Class order

```
class order
{
private:
    string famname;
    double timein;
    double timeout;
public:
    order(string name,double in,double out)
    {
        famname=name;
        timein=in;
        timeout=out;
    }
    void shownode()
    {
        //cout<<fixed();
        cout <<setprecision(2)<<fixed;
        cout<<famname<<" family go to shop between "<<timein<<" and "<<timeout<<endl;
    }
    double returnin()
    {
        return timein;
    }
};
```

3.2 Class building

```
class building{
private:
    int time[36];
    string name;
    double open;
    double close;
    int people;
    vector<order> orderlist;
    vector<int> gotoshop;
public:
    building(string);
    building(string,int);
    building(string,int,double,double);
    ~building();
    void addorder(string nam,double in,double out)
    {
        orderlist.push_back(order(nam,in,out));
        for(int i=0;i<orderlist.size();i++)
        {
            for(int j=i+1;j<orderlist.size();j++)
            {
                if(orderlist[i].returnin()>orderlist[j].returnin())
                {
                    swap(orderlist[i],orderlist[j]);
                }
            }
        }
    }
    void showorder()
    {
        for(int i=0;i<orderlist.size();i++)
        {
            cout<<"     "<<i+1<<") ";
            orderlist[i].shownode();
        }
    }
    int getnumpeople()
    {
        return gotoshop.size();
    }
    void checkadd(int x)
    {
        for(int i=0;i<gotoshop.size();i++)
        {
            if(gotoshop[i]==x)
            {
                return;
            }
        }
        gotoshop.push_back(x);
        return;
    }
    void operator+(int x)
    {
        ++time[x];
    }
```

```
    int getpeople()
    {
        return people;
    }
    double getopen()
    {
        return open;
    }
    double getclose()
    {
        return close;
    }
    int gettime(int x)
    {
        return time[x];
    }
    void shownode();
    void showtime();
    string getname(){
        return name;
    }
};
```

3.3 Class family

```
class  family{
        int time[36];
        string name;
        family* next;
    public:
        family(string);
        void book(int,int);
        void display();
        string getname(){
            return name;
        }
        int gettime(int i){
            return time[i];
        }
        void show_node();
        void insert(family*&);
        family* move_next();
        ~family();
};
```

3.4 Class town

```
class town{
    private:
        int size;
        family*head;
    public:
        void add_family(family*&);
        void show_all();
        family* find_fam(string);
        int getsize(){
            return size;
        }
        town();
        ~town();
};
```

4. ข้อกำหนด

4.1 Linked list

ใช้ Linked list town เพื่อเก็บ node family

```
family *pt;
town city;

pt=city.find_fam(fam_name);
if(pt==NULL){
    pt=new family(fam_name);
    city.add_family(pt);
}
else check=1;
```

4.2 Sorting Algorithm

ใช้ Bubble sort ในการเก็บลำดับการเข้าร้านค้าของแต่ละครอบครัว โดยจะเรียงเป็นลำดับตามเวลาที่เข้าก่อนอยู่ใน vector order

```
void addorder(string nam,double in,double out)
{
    orderlist.push_back(order(nam,in,out));
    for(int i=0;i<orderlist.size();i++)
    {
        for(int j=i+1;j<orderlist.size();j++)
        {
            if(orderlist[i].returnin()>orderlist[j].returnin())
            {
                swap(orderlist[i],orderlist[j]);
            }
        }
    }
}
```

4.3 Class with constructor

4.3.1 Class family เป็น class ของ node family เก็บ ชื่อครอบครัว ตารางเวลา และ address ของครอบครัวถัดไป

```
family::family(string s){
            name=s;
            for(int i=0;i<36;i++){
                time[i]=0;
            }
            next=NULL;
}
```

4.3.2 Class town เป็น linked list ที่เก็บ node family

```
town::town(){
    size=0;
    head=NULL;
}
```

4.3.3 Class building เป็น class ของร้านค้าต่าง ๆ ภายในเมือง

```
building::building(string nam,int peo,double ope,double clo)
{
    name=nam;
    people=peo;
    open=ope;
    close=clo;
    int op=open*2-8;
    system("CLS");
    cout<<"Shop "<<name<<" register successful\n";
    if(open!=(int)open)
    {
        op++;
    }
    int cl=close*2-8;
    if(close!=(int)close)
    {
        cl++;
    }
    for(int i=0;i<36;i++)
    {
        if(i>=op&&i<cl)
            time[i]=0;
        else
            time[i]=people;
    }
    shownode();
}
```

4.3.4 Class order เป็น class ของครอบครัวที่ลงทะเบียนในแต่ละร้านค้า

```
class order
{
private:
    string famname;
    double timein;
    double timeout;
public:
    order(string name,double in,double out)
    {
        famname=name;
        timein=in;
        timeout=out;
    }
    void shownode()
    {
        //cout<<fixed();
        cout <<setprecision(2)<<fixed;
        cout<<famname<<" family go to shop between "<<timein<<" and "<<timeout<<endl;
    }
    double returnin()
    {
        return timein;
    }
};
```

4.4 Polymorphism

4.4.1 ใช้ overloading constructor ในการประกาศ class building ทั้ง 3 รูปแบบ

```
building(string);
building(string,int);
building(string,int,double,double);
```

4.4.2 ใช้ overloading operator ในการจองเวลาของแต่ละร้านค้าใน class building

```
void operator+(int x)
{
    ++time[x];
}

vector<building> LL;

for(int j=entt;j<outt;j++)
{
    LL[i]+j;
    pt->book(j,i+1);

}
```

4.5 Exception Handling

ใช้ในการดัก input ที่ไม่ถูกต้องทั้งหมด เช่น เป็นตัวอักษร เป็นจำนวนลบ หรือเวลาปิดมาก่อนเวลาเปิด เป็นต้น (ในตัวอย่างคือดัก Input ในการจองเวลา)

```
try
{
    avilableshop(LL[tt]);
    cout<<"Input time when you will enter: ";
    ent_h=100;
    ent_mn=100;
    out_h=100;
    out_mn=100;
    if(!scanf("%d.%d",&ent_h,&ent_mn))
    {
        throw 1;
    }
    if(ent_h>24||ent_h<0||ent_mn<0||ent_mn>=60)
        throw 1;
    if(ent_h>22||ent_h<4||ent_mn<0||ent_mn>=60)
        throw 2;
    if(ent_h==22&&ent_mn!=0)
        throw 2;
    if(ent_mn!=30&&ent_mn!=0)
        throw 4;
    cout<<"Input time when you will leave: ";
    if(!scanf("%d.%d",&out_h,&out_mn))
    {
        throw 1;
    }

    if(out_h>24||out_h<0||out_mn<0||out_mn>=60)
        throw 1;
    if(out_h>22||out_h<4||out_mn<0||out_mn>=60)
        throw 2;
    if(out_h==22&&out_mn!=0)
        throw 2;
    if(out_mn!=30&&out_mn!=0)
        throw 4;
    if((out_h*60)+out_mn<=(ent_h*60)+ent_mn)
        throw 3;
    ent = (double)ent_h+(double)(ent_mn*0.01);
    out = (double)out_h+(double)(out_mn*0.01);
    entt=ent*2-8;
    if(ent!=(int)ent)
        entt++;
    outt=out*2-8;
    if(out!=(int)out)
        outt++;
    for(i=entt;i<outt;i++){
        if(pt->gettime(i)!=0){
            throw 5;
            break;
        }
    }

    if (res.compare(LL[i].getname())==0)
    {
        for(int j=entt;j<outt;j++)
        {
            if(LL[i].gettime(j)>=LL[i].getpeople())
                throw 0;
        }
```

```
catch(int err)
{
    if(err==0)
    {
        cout<<"This time is full\n";
        cout<<"Press F for select shop again, any key for add time again\n     ";
        returntime=_getch();
        if(returntime=='F'||returntime=='f')
        {
            cin.clear();
            cin.ignore(5000,'\n');
            system ("CLS");
            break;
        }
        //cin.clear();
          //  cin.ignore(5000,'\n');
            system ("CLS");
    }
    else if(err==1)
    {
        cout<<"Invalid input\n";
        cout<<"Press F for select shop again, any key for add time again\n     ";
        returntime=_getch();
        if(returntime=='F'||returntime=='f')
        {
            cin.clear();
            cin.ignore(5000,'\n');
            system ("CLS");
            break;
        }
      //  cin.clear();
         // cin.ignore(5000,'\n');
            system ("CLS");
    }
```

5. ข้อจำกัด

5.1 ไม่มีการบันทึกข้อมูลในโปรแกรมไว้ หากปิดโปรแกรมข้อมูลจะหายและต้องกรอกข้อมูลใหม่ทั้งหมด

5.2 ตารางเวลาในการจองเป็นช่องละ 30 นาที ซึ่งอาจไม่ละเอียดเพียงพอ

5.3 ไม่มีระบบป้องกันหรือยืนยันตัวตน ทำให้การจองไม่มีความปลอดภัย อาจมีคนอื่นมาใช้สิทธิเราได้หรือ จองเล่นจนร้านค้าเต็ม

5.4 รับ Input ได้มากสุด 5000 ตัวและไม่สามารถรับชื่อที่มีช่องว่าง (Spacebar) ได้

5.5 ไม่สามารถยกเลิกการจองและเวลาเคอร์ฟิวไม่สามารถเปลี่ยนแปลงได้

5.6 ไม่ได้จำกัดว่าครอบครัวหนึ่งสามารถจองได้กี่ร้านค้าและร้านค้าละกี่ชั่วโมง

5.7 ไม่มีระบบล้างตารางเวลาทั้งหมด หากต้องการ Reset ต้องปิดโปรแกรมแล้วเปิดใหม่