ภาคเรียนที่ 2 ปีการศึกษา 2562

Business fight with Covid-19

In theme Covid-19

เสนอ:

อาจารย์ ดร.มิ่งมานัส ศิวรักษ์

จัดทำโดย:

นายติณห์ ไชยเสนา 6213199

นายปณต เล็กเจริญ 6213203

น.ส.ภคพร พิพัฒนสุขมงคล 6213207

คำนำ

โครงงาน เรื่อง Business fight with Covid-19 เป็นส่วนหนึ่งของรายวิชา วศคพ 112 เทคนิคการเขียนโปรแกรม (EGCO 112 PROGRAMMING TECHNIQUES) หลักสูตรวิศวกรรมศาสตร์บัณฑิต สาขาวิชาวิศวกรรมคอมพิวเตอร์ มหาวิทยาลัยมหิดล ภาคเรียนที่ 2 ปีการศึกษา 2562

หากมีผิดพลาดประการใดทางคณะผู้จัดทำก็ขออภัยมา ณ ทีนี้ด้วย

คณะผู้จัดทำ

Contents

ไอเดียและการทำงานของโปรแกรม 1

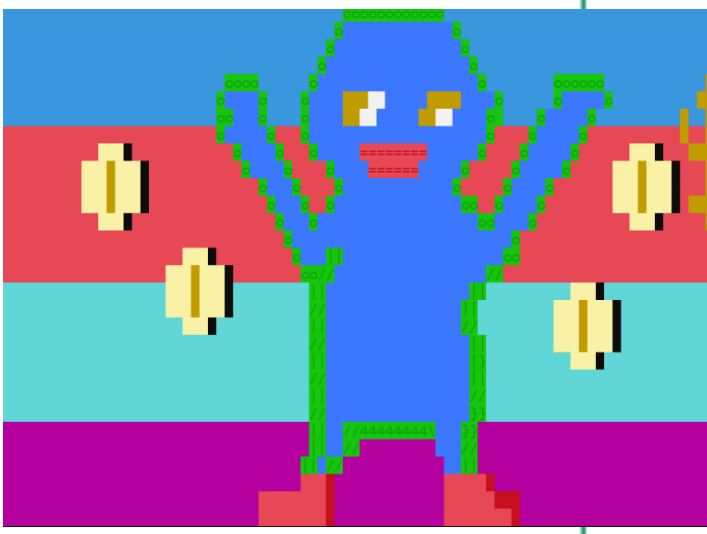
Class template 2

Requirement 9

ข้อจำกัดโปรแกรม 14

ไอเดียและการทำงานของโปรแกรม

เนื่องจากสถานะการการแพร่ระบาดของไวรัส Covid-19 เราได้เห็นความลำบากในการใช้ชีวิตของผู้คน รวมไปถึงยังส่งผลกระทบต่อธุรกิจ



เราจึงได้จัดทำเกมส์ที่มีจุดมุ่งหมายในการบริหารธุรกิจ พร้อมกับการเอาชีวิตรอดไปด้วย โดยตัวละครหลักของเราในเกมส์นี้เป็นเจ้าของธุรกิจที่เคยร่ำรวย ซึ่งได้รับผลกระทบจากการแพร่ระบาดของไวรัส Covid-19 จนทำให้ต้องปิดตัวธุรกิจลง และสร้างธุรกิจทำเครื่องดื่ม Soda ใหม่ ด้วยเงินที่เหลืออยู่ 40,000 และต้องเอาชีวิตรอดไป 30 วันจึงจะได้รับการช่วยเหลือ

วิธีการเล่น

ใช้ปุ่มลูกศรขึ้นและลงบนแป้นพิมพ์เพื่อบังคับตัวชี้ จากนั้นกด enter เพื่อเลือกตัวเลือกที่ต้องการ

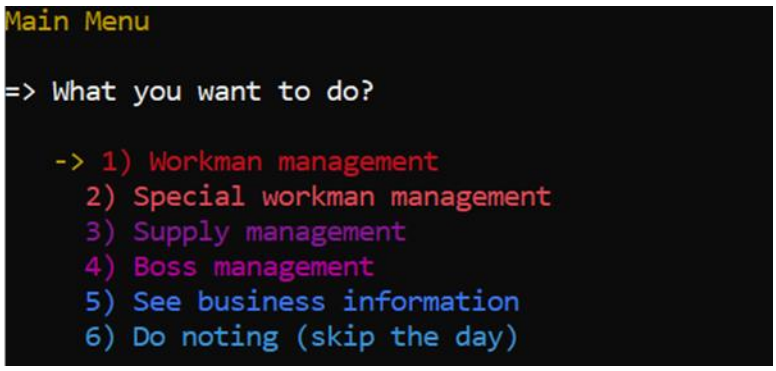
เมื่อเริ่มเกมมาผู้เล่นจะได้รับบทเป็น Boss ผู้เป็นเจ้าของธุรกิจและจะได้รับเงินเริ่มต้น 40,000 บาท โดยผู้เล่นต้องบริหารเงิน, HP (พลังชีวิต) และ MP (สุขภาพจิต) โดยผู้เล่นจะต้องบริหารการเงินเพื่อให้เพียงพอในการเอาชีวิตรอด ผู้เล่นจะชนะ เมื่อเอาชีวิตรอดจนถึงวันที่ 30

เกมส์นี้ผู้เล่นจะแพ้ได้ 2 วิธี คือ

1. ค่าพลังชีวิตหรือ hp เหลือ 0 จากการติด Covid ซึ่งจะสุ่มเป็นเปอร์เซ็น(1-100)ในแต่ละวัน เมื่อติด Covid จะส่งผลต่อค่า HP หรือค่าพลังชีวิตของผู้เล่นจะลดวันละ 20 หน่วย หากพลังชีวิตหมด ผู้เล่นจะป่วยตาย

2. ค่าสุขภาพจิตหรือ MP เหลือ 0 โดยในแต่ละวัน ค่าสุขภาพจิตจะค่อยๆลดวันละ 10 หน่วย และหากติด Covid MP ของผู้เล่น จะลดลงเพิ่มขึ้นเป็น 20 หน่วยต่อวัน เมื่อค่าสุขภาพจิตหมด ตัวละครของผู้เล่นจะทำการฆ่าตัวตาย

หน้า main menu จะมีหัวข้อต่างๆในการบริหารในแต่ละวัน ได้แก่



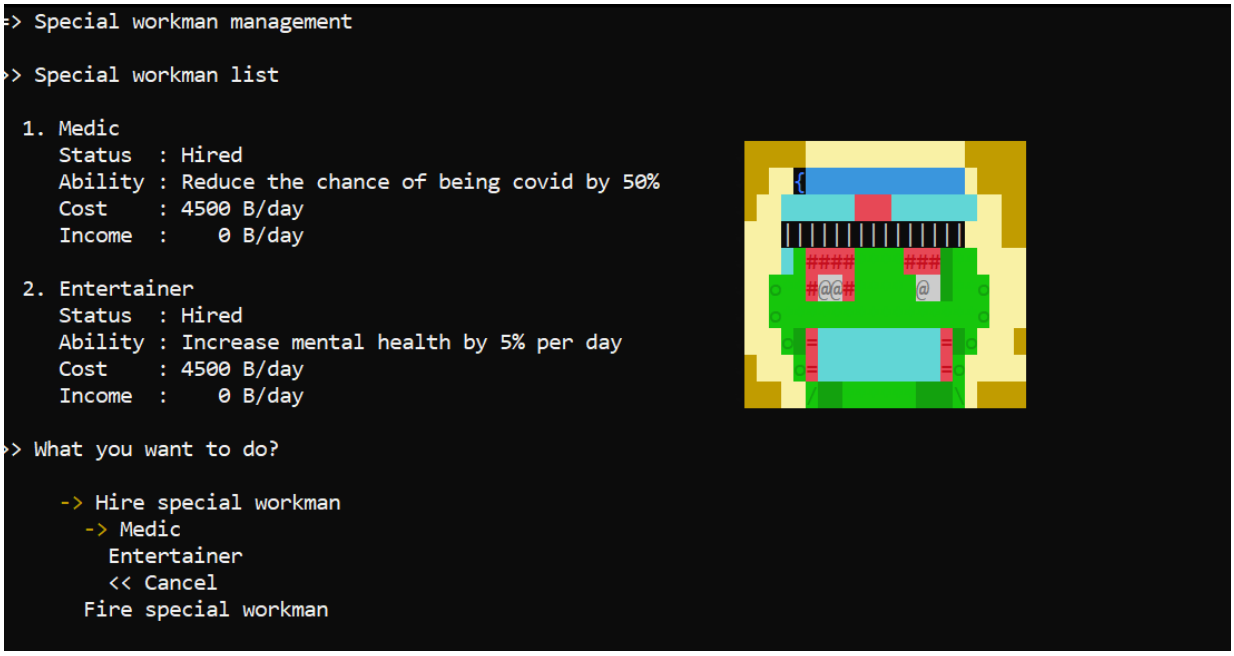
1. Workman management เป็นการบริหารการจ้างคนงาน ซึ่งคนงานแต่ละระดับมีค่าจ้างและเงินที่สามารถทำได้และวัตถุดิบที่ใช้ในการผลิต แตกต่างกันไป โดยผู้เล่นสามารถเลือกระดับของพนักงาน และตั้งชื่อพนักงานได้

ผู้เล่นสามารถไล่พนักงานออกได้ และเนื่องจากพนักงานมีโอกาศในการติด Covid-19 ทำให้หากพนักงานคนนั้นติด Covid พนักงานคนนั้นจะถูกไล่ออก

2. Special workman management เป็นตัวเลือกในการจ้างพนักงานพิเศษ ซึ่งจะมี 2 คน คือ

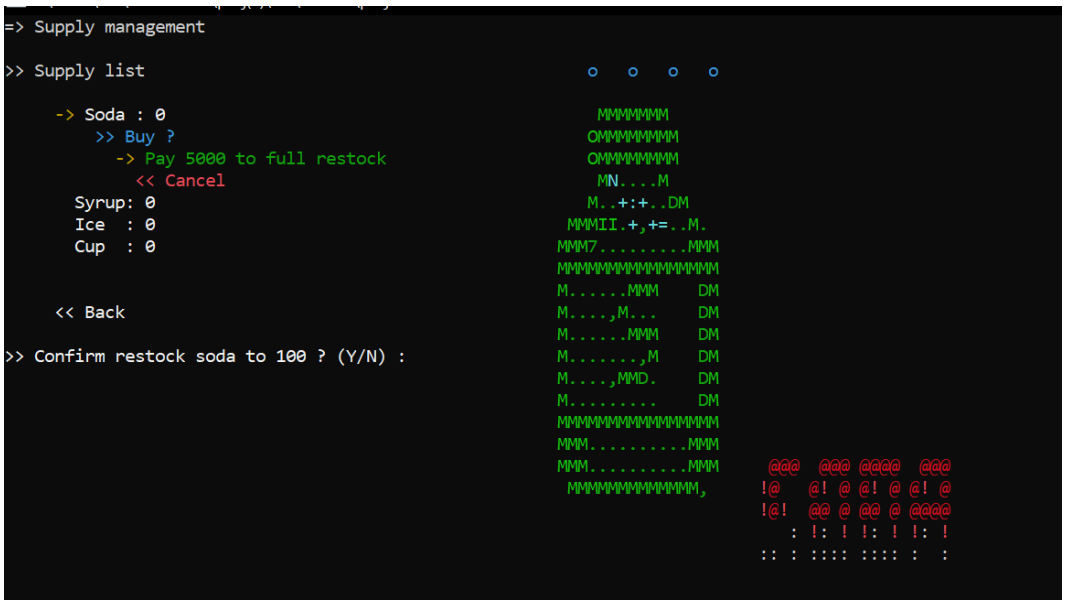
1.medic มีความสามารถคือ สามารถ ลดโอกาสการติด Covid-19 ได้

2.Entertainer มีความสามารถในการเพิ่ม MP หรือค่าสุขภาพจิต โดยเพิ่มวันละ 5 MP

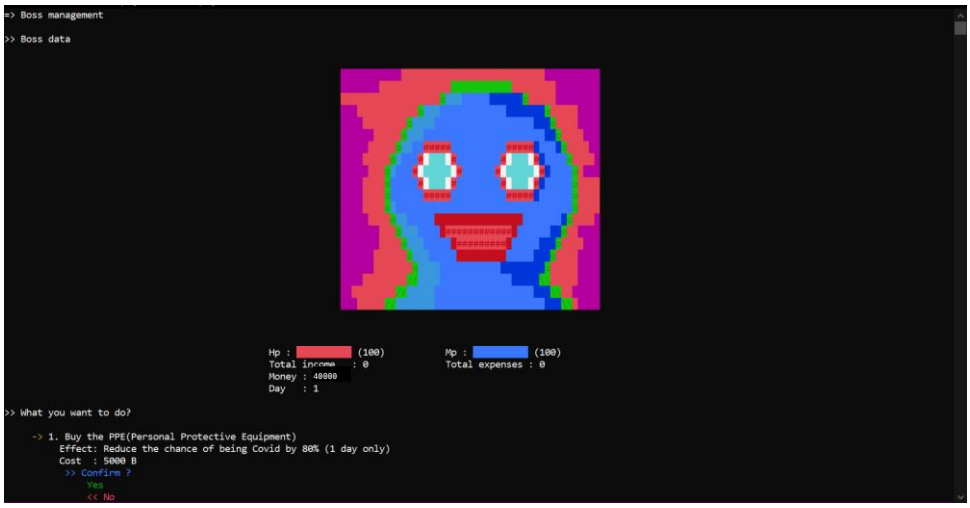


โดยผู้เล่นสามารถจ้างและไล่พนักงานพิเศษออกได้เช่นเดียวกับพนักงานปกติ แต่มีเพียงพนักงานพิเศษเท่านั้นที่จะไม่มีโอกาสติด Covid-19

3. Supply management เป็นตัวเลือกที่ใช้ในการเติมวัตถุดิบ โดยในการเติม 1 ครั้ง จะ restock วัตถุดิบให้เต็ม หรือเท่ากับ 100 วัตถุแต่ละชนิด จะมีค่าใช้จ่ายในการ restock ไม่เท่ากัน



4. Boss management เป็นการเลือกซื้อตัวเลือกพิเศษที่เกี่ยวข้องกับตัวผู้เล่นเอง โดยมีอยู่ 3 อย่าง คือ



1. Buy the PPE เป็นตัวเลือกที่จะช่วยลดโอกาสในการติด Covid-19 ให้กับตัวผู้เล่นเอง และพนักงานทุกคน โดยลดไป80% ของโอกาสติดเดิม ซึ่งจะมีผลแค่1วันเท่านั้น

2. Cure yourself จะเป็นตัวเลือกที่จะเพิ่ม HP ของผู้เล่นจนเต็ม อีกทั้งยังลดอัตราการเสีย HP จากการติด Covid-19 ลง 50 %

3. Entertain yourself เป็นการเพิ่มค่า MP หรือค่าสุขภาพจิตให้เต็ม

5. See business information เป็นการแสดงข้อมูลต่างๆในการบริหาร

```
                                        Hp : [          ] (100)          Mp : [          ] (100)
                                        Total income   : 0               Total expenses : 0
                                        Money : 40000
                                        Day   : 1
                                        ----------------------------------------------------------------

>> Material

Soda : 0
Syrup: 0
Ice  : 0
Cup  : 0

-------------------------------------------------------------

>> Hired Workman list

   1. - Empty -

   2. - Empty -

   3. - Empty -

   4. - Empty -

-------------------------------------------------------------

>> Special workman status

  1. Medic
     Status  : Fired
     Ability : Reduce the chance of being covid by 50%
     Cost    : 4500 B/day
     Income  :    0 B/day

  2. Entertainer
     Status  : Fired
     Ability : Increase mental health by 5% per day
     Cost    : 4500 B/day
     Income  :    0 B/day
```

6.Do Nothing (Skip the day) เป็นการจบวัน ซึ่งจะแสดงการสรุปผลในแต่ละวัน

```
=> Skip the day

Summary :

----- Day 1 -----

>> Workman

Money -1300 --> Money left: 75700

Money +3000 --> Money left: 78700

- 16 Soda  --> Soda left: 84
- 20 Syrup --> Syrup left: 80
- 35 Ice   --> Ice left: 65
- 15 Cup   --> Cup left: 85

>> Special workman

Money -4500 --> Money left: 74200

Medic : Reduce the chance of being covid by 50%

>> Boss

Hp    0 --> hp left: 100
Mp  -10 --> mp left:  90

-----------------
```

-แสดงจำนวนวัน

-แสดงการเปลี่ยนแปลงของจำนวนเงิน

-แสดงการเปลี่ยนแปลงของจำนวนวัตถุดิบ

-แสดงผลของ special workman

-แสดงการเปลี่ยนแปลงของค่าพลังชีวิต และสุขภาพจิตของ boss

Class template

Class workman

```
class workman: public material
{
protected:
    string name;
    string rank;
    int cost;
    int income;
    int level;
    int covid_rate;
public:
    workman(string = "Norank", int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, string = "Noname");
    void operator = (workman);
    void operator += (string);
    virtual void display()
    {
        if(name == "Noname")
            cout << "> Lv." << level << '\n';
        else
            cout << "Name   : " << name << " (Lv." << level << ")\n";
        space(5);
        cout << "Rank   : " << rank << '\n';
        space(5);
        printf("Cost   : %4d\n", cost);
        space(5);
        printf("Income : %4d\n", income);
        space(5);
        printf(">> Material use\n");
        space(5);
        printf("Soda: %2d   Syrup: %2d   Ice: %2d   Cup: %2d\n\n", soda, syrup, ice, cup);
    }
    string name_get();
    int cost_get();
    int income_get();
    int covid_rate_get();
    bool covid_cal(int);
    ~workman();
};
```

Class material

```
using namespace std;
class material{
protected:
    int soda;
    int syrup;
    int ice;
    int cup;
public:
    material(int = 0, int = 0, int = 0, int = 0);
    int show_soda();
    int show_syrup();
    int show_ice();
    int show_cup();
    void restock_soda();
    void restock_syrup();
    void restock_ice();
    void restock_cup();
    void minus_soda(int );
    void minus_syrup(int );
    void minus_ice(int );
    void minus_cup(int );
    ~material();
};
```

Class Node

```
class node{
private:
    workman data;
    node *next;
public:
    node(workman);
    void insert(node *&);
    node *move_next();
    workman data_get();
    ~node();
};
node::node(workman x){
    data = x;
    next = NULL;
    cout << "adding\n";
}
void node::insert(node *&x){
    this->next = x;;
}
node* node::move_next(){
    return next;
}
workman node::data_get(){
    return data;
}
node::~node(){
}
#endif // node_h
```

Class ll

```
class ll{
private:
    node *head;
    int size;
public:
    ll();
    int size_get();
    bool is_empty();
    void add_node(workman &);
    void print_list();
    void name_display(int);
    void delete_node(int);
    int cost_get_all();
    int income_get_all();
    int soda_get_all();
    int syrup_get_all();
    int ice_get_all();
    int cup_get_all();
    void covid_display_all(int);
    //~ll();
};
```

Class special_workman

```
class special_workman:public workman{
private:
    double rate;
    string info;
    bool status;
public:
    special_workman(double = 0, string = "NOINFO", bool = false, int = 0, int = 0, string = "NORANK");
    void display(int = 0);
    bool status_get();
    void status_set(bool);
    ~special_workman();
};
special_workman::special_workman(double x, string fo, bool st, int co, int in, string ra):workman(ra, co, in){
    rate = x;
    info = fo;
    status = st;
}
```

Class boss

```
class boss{
private:
    int hp;
    int mp;
    int total_income;
    int total_expenses;
    int money;
    int day;
    int covid_rate;
    int hp_damage;
    int mp_damage;
    bool covid;
public:
    boss(int = 100, int = 100, int = 0, int = 0, int = 100000, int = 1, int = 10, int = 0, int = 10, bool = false);
    void display();
    void display_block(int);
    void hp_set(int);
    void hp_add(int);
    void mp_set(int);
    void mp_add(int);
    void hp_damage_set(int);
    void mp_damage_set(int);
    bool covid_get();
    void covid_cal(int);
    int money_get();
    void money_add(int);
    int hp_damage_get();
    int mp_damage_get();
    void day_plus();
    int day_get();
    ~boss();
};
```

Class item

```
class item{
private:
    bool avaliable;
public:
    item(bool = false);
    bool avaliable_get();
    void avaliable_set(bool);
    ~item();
};
item::item(bool a){
    avaliable = a;
}
bool item::avaliable_get(){
    return avaliable;
}
void item::avaliable_set(bool x){
    avaliable = x;
}
item::~item(){
}
#endif // item_h
```

Requirement

1. data structure: linked list

```
class node{
private:
    workman data;
    node *next;
public:
    node(workman);
    void insert(node *&);
    node *move_next();
    workman data_get();
    ~node();
};
node::node(workman x){
    data = x;
    next = NULL;
    cout << "adding\n";
}
void node::insert(node *&x){
    this->next = x;;
}
node* node::move_next(){
    return next;
}
workman node::data_get(){
    return data;
}
node::~node(){
}
#endif // node_h
```

```
struct node{
    workman data;
    node *next;
};
class ll{
private:
    node *head;
    int size;
public:
    ll();
    int size_get();
    bool is_empty();
    void add_node(workman &);
    void print_list();
    void name_display(int);
    void delete_node(int);
    int cost_get_all();
    int income_get_all();
    int soda_get_all();
    int syrup_get_all();
    int ice_get_all();
    int cup_get_all();
    void covid_display_all(int);
    //~ll();
};
ll::ll(){
    head = NULL;
    size = 0;
}
int ll::size_get(){
    return size;
}
bool ll::is_empty(){
    if(size == 0)
        return true;
    else
        return false;
}
```

```
void ll::add_node(workman &a){
    node *new_node = new node;
    new_node->data = a;
    new_node->next = NULL;
    if(size == 0){
        head = new_node;
    }
    else{
        node *t = head;
        for(int i = 0; i < size-1; ++i)
            t = t->next;
        t->next = new_node;
    }
    ++size;
}
void ll::print_list(){
    node *t = head;
    for(int i = 1; i <= size; ++i){
        printf("%3d. ", i);
        t->data.display();
        t = t->next;
    }
}
void ll::name_display(int n){
    node *t = head;
    for(int i = 1; i < n; ++i)
        t = t->next;
    cout << t->data.name_get() << '\n';
}
void ll::delete_node(int x){
    node *t = head;
    if(x == 1){
        head = head->next;
    }
    else{
        for(int i = 1; i <= x-2; ++i)
            t = t->next;
        t->next = t->next->next;
    }
    --size;
}
```

2. a sorting algorithm

```
void ll::bubble_sort(){
    int swapped, i;
    node *t;
    node *lptr = NULL;
    if (head == NULL)
        return;
    do
    {
        swapped = 0;
        t = head;
        while (t->next != lptr)
        {
            if (t->data.lv_get() > t->next->data.lv_get())
            {
                node *a = t;
                node *b = t->next;
                workman temp = a->data;
                //swap
                a->data = b->data;
                b->data = temp;
                swapped = 1;
            }
            t = t->next;
        }
        lptr = t;
    }
    while (swapped);
}
```

3. class with constructor

Special workman

```
using namespace std;
class special_workman:public workman{
private:
    double rate;
    string info;
    bool status;
public:
    special_workman(double = 0, string = "NOINFO", bool = false, int = 0, int = 0, string = "NORANK");
    void display(int = 0);
    bool status_get();
    void status_set(bool);
    ~special_workman();
};
special_workman::special_workman(double x, string fo, bool st, int co, int in, string ra):workman(ra, co, in){
    rate = x;
    info = fo;
    status = st;
}
void special_workman::display(int x){
    space(x-5); cout << rank << '\n';
    space(x); cout << "Status  : ";
    if(status == true) cout << "Hired";
    else cout << "Fired";
    cout << '\n';
    space(x); cout << "Ability : " << info << '\n';
    space(x); printf("Cost    : %4d B/day\n", cost);
    space(x); printf("Income  : %4d B/day\n", income);
}
bool special_workman::status_get(){
    return status;
}
void special_workman::status_set(bool x){
    status = x;
}
special_workman::~special_workman(){
}
#endif // special_workman_h
```

Workman บางส่วน

```
class workman: public material
{
protected:
    string name;
    string rank;
    int cost;
    int income;
    int level;
    int covid_rate;
public:
    workman(string = "Norank", int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, string = "Noname");
    void operator = (workman);
    void operator += (string);
    virtual void display()
    {
        if(name == "Noname")
            cout << "> Lv." << level << '\n';
        else
            cout << "Name   : " << name << " (Lv." << level << ")\n";
        space(5);
        cout << "Rank   : " << rank << '\n';
        space(5);
        printf("Cost   : %4d\n", cost);
        space(5);
        printf("Income : %4d\n", income);
        space(5);
        printf(">> Material use\n");
        space(5);
        printf("Soda: %2d   Syrup: %2d   Ice: %2d   Cup: %2d\n\n", soda, syrup, ice, cup);
    }
    string name_get();
    int cost_get();
    int income_get();
    int covid_rate_get();
    bool covid_cal(int);
    ~workman();
};
```

```
workman::workman(string ra, int co, int in, int lv, int co_ra, int so, int sy, int ic, int cu, string na):material(so, sy, ic, cu)
{
    rank = ra;
    cost = co;
    income = in;
    level = lv;
    covid_rate = co_ra;
    name = na;
}
workman::~workman()
{
}
void workman::operator = (workman w)
{
    //workman
    cost = w.cost;
    income = w.income;
    rank = w.rank;
    name = w.name;
    level = w.level;
    covid_rate = w.covid_rate;
    name = w.name;
    //material
    soda = w.soda;
    syrup = w.syrup;
    ice = w.ice;
    cup = w.cup;
}
void workman::operator += (string n)
{
    name = n;
}
```

Material บางส่วน

```
using namespace std;
class material{
protected:
    int soda;
    int syrup;
    int ice;
    int cup;
public:
    material(int = 0, int = 0, int = 0, int = 0);
    int show_soda();
    int show_syrup();
    int show_ice();
    int show_cup();
    void restock_soda();
    void restock_syrup();
    void restock_ice();
    void restock_cup();
    void minus_soda(int );
    void minus_syrup(int );
    void minus_ice(int );
    void minus_cup(int );
    ~material();
};
```

```
material::material(int so, int sy, int ic, int cu)
{
    soda = so;
    syrup = sy;
    ice = ic;
    cup = cu;
}
material::~material() {}

int material::show_cup()
{
    return cup;
}
void material::restock_cup()
{
    cup = 100;
}
void material::minus_cup(int cu)
{
    cup -= cu;
}
```

Boss บางส่วน

```
using namespace std;
class boss{
private:
    int hp;
    int mp;
    int total_income;
    int total_expenses;
    int money;
    int day;
    int covid_rate;
    int hp_damage;
    int mp_damage;
    bool covid;
public:
    boss(int = 100, int = 100, int = 0, int = 0, int = 100000, int = 1, int = 10, int = 0, int = 10, bool = false);
    void display();
    void display_block(int);
    void hp_set(int);
    void hp_add(int);
    void mp_set(int);
    void mp_add(int);
    void hp_damage_set(int);
    void mp_damage_set(int);
    bool covid_get();
    void covid_cal(int);
    int money_get();
    void money_add(int);
    int hp_damage_get();
    int mp_damage_get();
    void day_plus();
    int day_get();
    ~boss();
};
boss::boss(int h, int m, int ti, int te, int mo, int d, int cr, int hd, int md, bool c)
{
    hp = h;
    mp = m;
    total_income = ti;
    total_expenses = te;
    money = mo;
    day = d;
    covid_rate = cr;
    hp_damage = hd;
    mp_damage = md;
    covid = c;
}
boss::~boss()
{
}
void boss::hp_add(int h)
{
    hp += h;
    if(hp > 100)
        hp = 100;
    if(hp < 0)
        hp = 0;
    printf("Hp ");
    if(h > 0)
        printf("+");
    printf(" %3d --> hp left: %3d\n", h, hp);
}
bool boss::covid_get()
{
    return covid;
}
```

4. polymorphism

```
protected:
    string name;
    string rank;
    int cost;
    int income;
    int level;
    int covid_rate;
public:
    workman(string = "Norank", int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, int = 0, string = "Noname");
    void operator = (workman);
    void operator += (string);
    virtual void display()
    {
        if(name == "Noname")
            cout << "> Lv." << level << '\n';
        else
            cout << "Name   : " << name << " (Lv." << level << ")\n";
        space(5);
        cout << "Rank   : " << rank << '\n';
        space(5);
        printf("Cost   : %4d\n", cost);
        space(5);
        printf("Income : %4d\n", income);
        space(5);
        printf(">> Material use\n");
        space(5);
        printf("Soda: %2d   Syrup: %2d   Ice: %2d   Cup: %2d\n\n", soda, syrup, ice, cup);
    }
```

5. Exception handling

In main

```
try
{
    cin >> lv;
    if(cin.fail())
        throw 0;
    if(lv != 99 && (lv < 0 || lv > 4))
        throw 1;
}
catch(int e)
{
    if(e == 0)
    {
        cout << "Cin fail, please enter number\n";
        press_any_key(25, 40);
        workman_management(1);
    }
    if(e == 1)
    {
        cout << "Number out of range, please enter 1-4 or 99\n";
        press_any_key(25, 40);
        workman_management(1);
    }
}
catch(exception &e)
{
    cout << "-> Exception: " << e.what() << '\n';
    cout << "Return to menu\n";
    menu(1);
}
```

```
ShowConsoleCursor(0);
if(lv == 99)
{
    delaycout("Return to workman management\n", 50);
    system("cls");
    workman_management(1);
    ShowConsoleCursor(0);
}
delaycout("Please enter workman name : ", 50);
ShowConsoleCursor(1);
try
{
    cin >> name;
}
catch(exception &e)
{
    cout << "-> Exception: " << e.what() << '\n';
    cout << "Return to menu\n";
    menu(1);
}
ShowConsoleCursor(0);
```

ข้อจำกัดโปรแกรม

1. หากกดตัวอื่นที่ไม่ใช่ลูกศรชี้ ตัวเลือกจะค้าง อาจต้องกด enter 1 ครั้งเพื่อให้สามารถเลื่อนลูกศรได้ปกติ
2. ในบางกรณีเช่น การทำรายการไม่สำเร็จ อาจทำให้ตัวชี้ค้าง ต้องกด enter 1 ครั้งจึงสามารถเลื่อนลูกศรได้
3. ใส่ข้อความได้แค่ภาษาอังกฤษและตัวเลขเท่านั้น
4. ตั้งชื่อ workman ได้แค่ภาษาอังกฤษและตัวเลขเท่านั้น
5. Total income และ Total cost แสดงแค่กำไรและค่าจ้างจาก workman และ special workman เท่านั้น
6. ไม่สามารถข้ามเนื้อเรื่องตอนเริ่มต้นได้
7. หาก Game over ต้องเริ่มใหม่ตั้งแต่ต้น
8. หากใส่ตัวอักษร ลงใน lv. Workman เกมจะถูกปิด