REPUBLIC OF CAMEROON

Peace- Work-Fatherland

REPUBLIQUE DU CAMEROUN

Paix-Travail-Patrie

THE UNIVERSITY OF BAMENDA

UNIVERSITE DE BAMENDA

THE COLLEGE OF TECHNOLOGY

(COLTECH)

ECOLE DE TECHNOLOGIE

COMPUTER ENGINEERING

SOFTWARE ENGINEERING

Object Oriented Programming C++

PRESENTED BY:

MBAH LESKY TAGWANG

UBA21PB015 (Level 300)

2022/2023

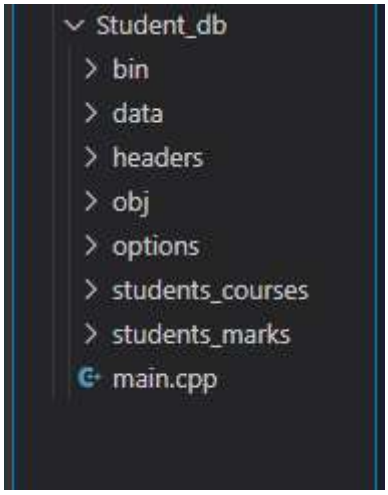
OUTLINE

Student Marks Database

Functions

- Register student to file
- Add new course to file
- Add course to department (SWE, CNSM, EEE)
- Add option and add courses to option
- Register courses under students
- Record student marks to file
- Search courses and students

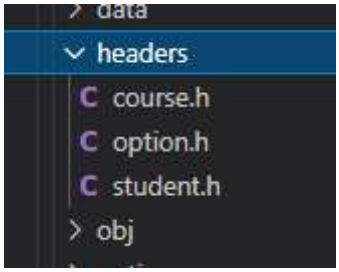
Project Directory



Code:

Headers

These header files contain the classes their properties, methods and other functions



A. Student Class (in student.h)

```
#ifndef STUDENT_H
#define STUDENT_H
#include <iostream>
#include <fstream>
#include <string.h>

using namespace std;

enum Gender {Male, Female, Other};

string genderPrint(Gender gender){
    switch(gender){
        case 0: return "Male";
        case 1: return "Female";
        case 2: return "Other";
        default: exit(1);
    }
}
```

- An enumerated list stores the different genders an a function returns the genders as strings

```cpp
class Student {
    public:
    string firstName;
    string secondName;
    string fullName;
    string matricule;
    Gender gender;
    string option;

    Student(string fname, string sname, string mat, Gender gen, string optCode){
        firstName = fname;
        secondName = sname;
        matricule = mat;
        gender =  gen;
        fullName = fname + " " + sname;
        option = optCode;

        // writing data to file
        ofstream studentsInfo, studentData;
        studentsInfo.open("data/students_info.csv", ios::app);
        if(!studentsInfo.is_open()){
            cout << "Error adding student\n";
            exit(1);
        }

        studentsInfo << firstName << "," << secondName << "," << matricule << "," << genderPrint(gender) << option << endl;
        studentsInfo.close();

        cout << fullName << " Successfully Added\n";
    }

    void registerDepartmentalCourses(){ ...

    void recordMarks(){ ...

    void setGPA(float value){ ...

    float getGPA(){
        return GPA;
    }

    private:
    float GPA;

};
```

- The Student class as some public properties, a contructor, some methods, a private property and a get and set methods

Methods:

Register student courses based on department or Option

```
void registerDepartmentalCourses(){

    // openning the file containing courses
    ifstream optionCourses;
    ofstream studentCourses;
    optionCourses.open("options/" + option + ".csv");
    studentCourses.open("students_courses/" + matricule + ".csv", ios::out);

    if(!optionCourses.is_open() || !studentCourses.is_open()){
        cout << "Error registering courses\n";
        exit(1);
    }

    // storing the courses in a list and registering under the student
    string course[1000];
    int i = 0, j;
    while(getline(optionCourses, course[i])){
        i++;
    }

    for (j = 3; j<i; j++){
        studentCourses << course[j] << endl;
    }

    cout << "registered courses\n";
    studentCourses.close();
    optionCourses.close();

}
```

- The courses from are read from the students option (stored in the options directory)

Record students marks for registered courses

```
void recordMarks(){
    ifstream studentCourses;
    ofstream studentMarks;
    studentCourses.open("students_courses/" + matricule + ".csv");
    studentMarks.open("students_marks/" + matricule + ".csv", ios::out);
    if(!studentCourses.is_open() || !studentMarks.is_open()){
        cout << "Error recording marks";
        exit(1);
    }
    string course;
    int cv, totalCredit = 0;
    float twa = 0, caMark = 0, examMark = 0, examtemp, catemp, totalMark, twp = 0;
    while(getline(studentCourses, course)){

        cout << "Enter marks for ";
        int i;
        for (i = 0; i < course.length() - 1; i++){
            if(course[i] == ','){
                i++;
                break;
            }
        }
        for (; i < course.length(); i++){
            cout << course[i];
            if(course[i] == ',') break;
        }
        cout << "\nCA Marks: ";
        cin >> catemp;
        cout << "Exam Mark: ";
        cin >> examtemp;
        cv = (int)course[course.length() - 1];
        examMark += examtemp;
        caMark += catemp;
        totalMark = caMark + examMark;
        float course_gp = totalMark/4;
        float course_wp = course_gp * cv;
        twa += course_wp;
        totalMark = examMark + caMark;
        totalMark/25;
        totalCredit += cv;
        studentMarks << course << "," << catemp << "," << examtemp << endl;
    }
        setGPA(twa/totalCredit);
        studentMarks << endl << "GPA: " << getGPA() << endl;
    cout << "Marks uploaded" << endl;
}
```

Get and Set methods for the private property

```
    void setGPA(float value){
        GPA = value;
    }

    float getGPA(){
        return GPA;
    }

    private:
    float GPA;

};
```

Functions: Adding a new student

```
void addStudent(){
    string fname, sname, matricule, option;
    int optionChoice, genderChoice;
    Gender gender;

    // getting data from user

    cout << "|-------------------------------------------------------------------|" << endl;
    cout << "|************************* Student Database **************************|" << endl;
    cout << "|                                                                   |" << endl;
    cout << "|\t 1. SWE\t\t2.CNSM \t  \t 3.EEE \t\t 4.EPE\t\t|" << endl;
    cout << "|                                                                   |" << endl;
    cout << "|****************************** Options ******************************|" << endl;
    cout << "|-------------------------------------------------------------------|" << endl << endl;

    do {
        cout << "Select an option: ";
        cin >> optionChoice;

        switch (optionChoice)
        {
        case 1:
            option = "SWE";
            break;
        case 2:
            option = "CNSM";
            break;
        case 3:
            option = "EEE";
            break;
        case 4:
            option = "EPE";
            break;
        default:
            break;
        }
    } while(optionChoice >= 5);

    cout << "Enter student's first name: ";
    cin >> fname;

    cout << "Enter student's second name: ";
    cin >> sname;

    cout << "Enter student's matricule: ";
```

- Gets input from user, then creates a Student object using the inputs and saves the data to a file

```
    } while(optionChoice >= 5);

    cout << "Enter student's first name: ";
    cin >> fname;

    cout << "Enter student's second name: ";
    cin >> sname;

    cout << "Enter student's matricule: ";
    cin >> matricule;

    cout << "1. Male\n" << "2. Female\n" << "3. Other\n";

    do {
        cout << "Select gender: ";
        cin >> genderChoice;

        switch (genderChoice)
        {
        case 1:
            gender = Male;
            break;
        case 2:
            gender = Female;
            break;
        case 3:
            gender = Other;
            break;
        }
    } while(genderChoice > 3);

    Student newStudent(fname, sname, matricule, gender, option);

}
#endif
```

B. Option class (option.h):

CA > Student_db > headers > C option.h > Option > Option(string, string, string)

```
#ifndef OPTION_H
#define OPTION_H
#include <iostream>
#include <fstream>
#include <string>

using namespace std;

class Option {
    public:
    string abbrev;
    string name;
    string school;

    Option(string abb, string title, string faculty){
        abbrev = abb;
        name = title;
        school = faculty;

        // writing data to file
        ofstream optionsInfo, optionFile;
        optionsInfo.open("data/options_info.csv", ios::app);
        optionFile.open("options/" + abbrev + ".csv");

        if(!optionsInfo.is_open() || !optionFile.is_open()){
            cout << "Error adding option\n";
            exit(1);
        }

        optionsInfo << abbrev << "," << name << "," << school << endl;
        optionsInfo.close();

        optionFile << "Name: " << name << "(" << abbrev << ")" << endl;
        optionFile << "School: " << school << endl << endl;
        optionFile.close();

        cout << name << " Successfully Added\n";
    }

    void addDepartmentalCourse(){ ...

};
#endif
```

Methods:

Add courses under an option(department)

```cpp
void addDepartmentalCourse(){

    // openning the file containing courses
    ifstream coursesInfo;
    coursesInfo.open("data/courses_info.csv");

    if(!coursesInfo.is_open()){
        cout << "Error adding course\n";
        exit(1);
    }

    // storing the courses in a list
    string course[1000];
    int i = 0, j, k;
    while(getline(coursesInfo, course[i])){
        i++;
    }

    for (j = 0; j < i; j++){
        cout << j+1 << ": ";
        for (k = 0; k < course[j].length(); k++){
            if (course[j][k] == ','){
                cout << "-";
                continue;
            }
            cout << course[j][k];
        }
        cout << endl;
    }

    // selecting courses
    cout << "Enter the number(s) of the courses you want to register\n" << "Seperated them with a spaces\n" << "Hit enter when you are done: ";
    int choices[i];
    j = 0;
    do {
        cin >> choices[j];
        j++;
    } while (getchar() != '\n');

    // registering courses under the option
    // openning the file containing courses
    ofstream optionData;
    optionData.open("options/" + abbrev + ".csv", ios::app);
```

```cpp
    // registering courses under the option
    // openning the file containing courses
    ofstream optionData;
    optionData.open("options/" + abbrev + ".csv", ios::app);

    if(!optionData.is_open()){
        cout << "Some Error occured\n";
        exit(1);
    }
    for (k = 0; k < j-1; k++){
        optionData << course[choices[k]] << endl;
    }

    cout << "added courses";
    optionData.close();
}

};
#endif
```

- Displays all the courses and lets the user select the courses to be registered for the option

C. Courses class (course.h)

CA > Student_db > headers > course.h > Course > Course(string, string, int)

```
#ifndef COURSE_H
#define COURSE_H
#include <iostream>
#include <fstream>

using namespace std;

class Course {
    public:
    string code;
    string title;
    int creditValue;

    Course(string id, string name, int cv){
        code = id;
        title = name;
        creditValue = cv;

         // writing data to file
        ofstream coursesInfo;
        coursesInfo.open("data/courses_info.csv", ios::app);

        if(!coursesInfo.is_open()){
            cout << "Error adding course\n";
            exit(1);
        }

        coursesInfo << code << "," << title << "," << creditValue << endl;
        coursesInfo.close();

        cout << title << " Successfully Added\n";
    }

    private:
    float ca_marks;
    float exam_marks;
    float total_marks;
    char grade;
    float GPA;
};
#endif
```

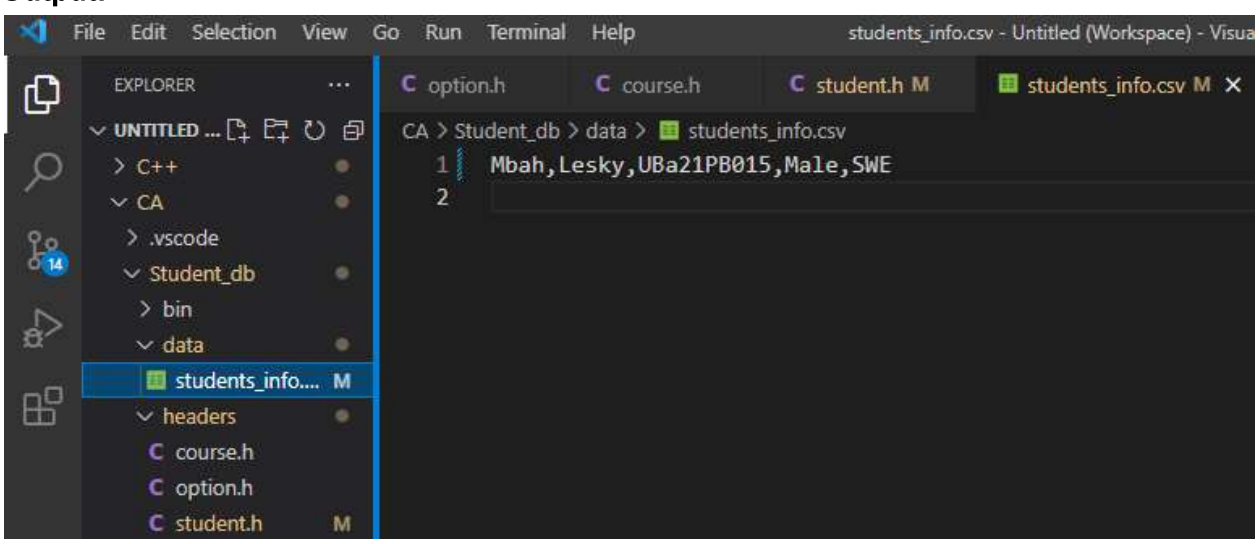
Outputs

- **Creating a new student**

Console

```
"D:\My Files\Documents\School-Work\C++\CA\Student_db\main.exe"
|-------------------------------------------------------------------------|
|************************** Student Database ***************************|
|                                                                         |
|        1. SWE          2.CNSM          3.EEE          4.EPE            |
|                                                                         |
|******************************** Options ******************************|
|-------------------------------------------------------------------------|

Select an option: 1
Enter student's first name: Mbah
Enter student's second name: Lesky
Enter student's matricule: UBa21PB015
1. Male
2. Female
3. Other
Select gender: 1
Mbah Lesky Successfully Added

Process returned 0 (0x0)   execution time : 21.941 s
Press any key to continue.
```

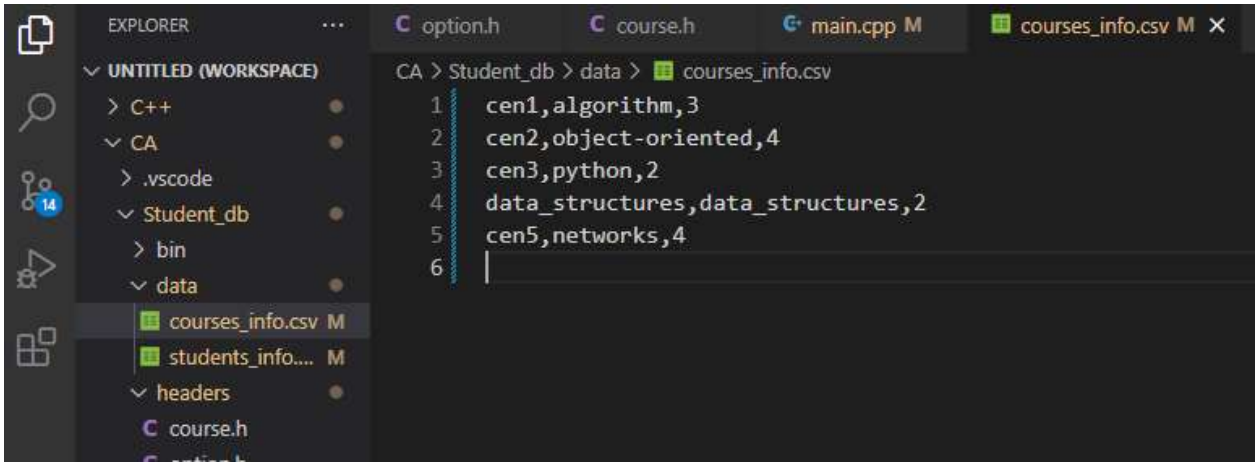
Output:



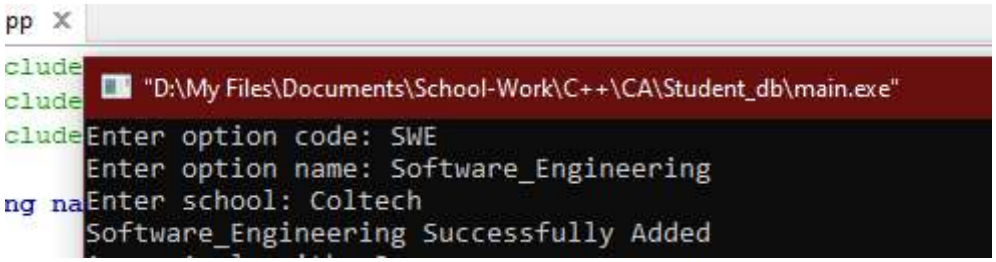
Information added to the students_info file

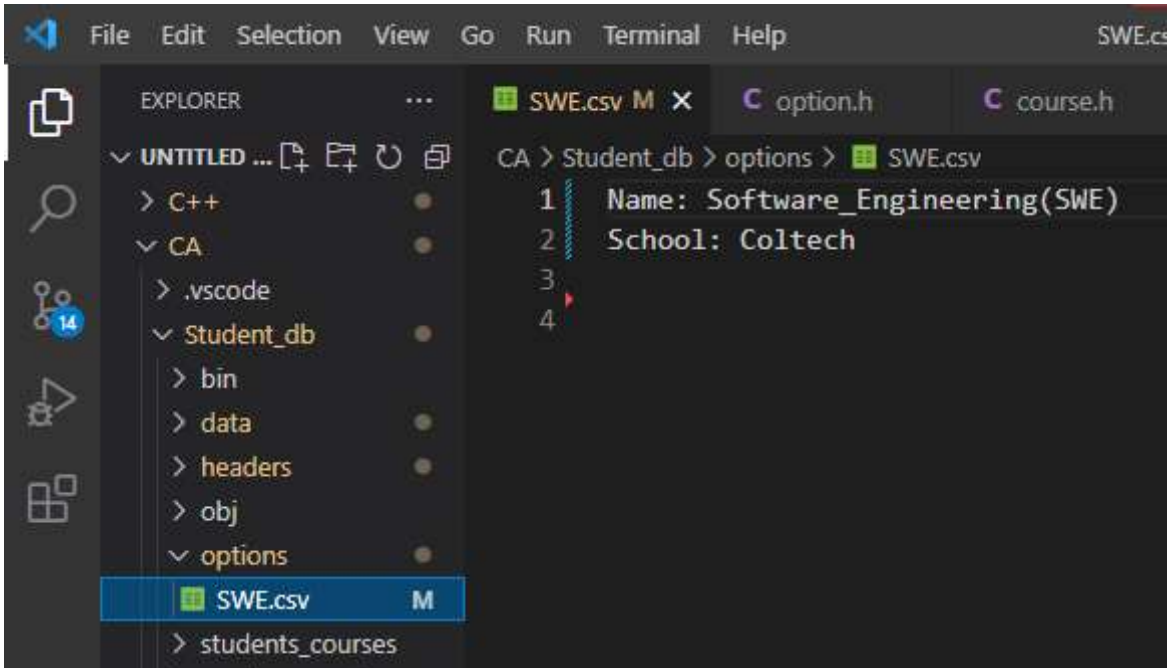
- **Adding courses**

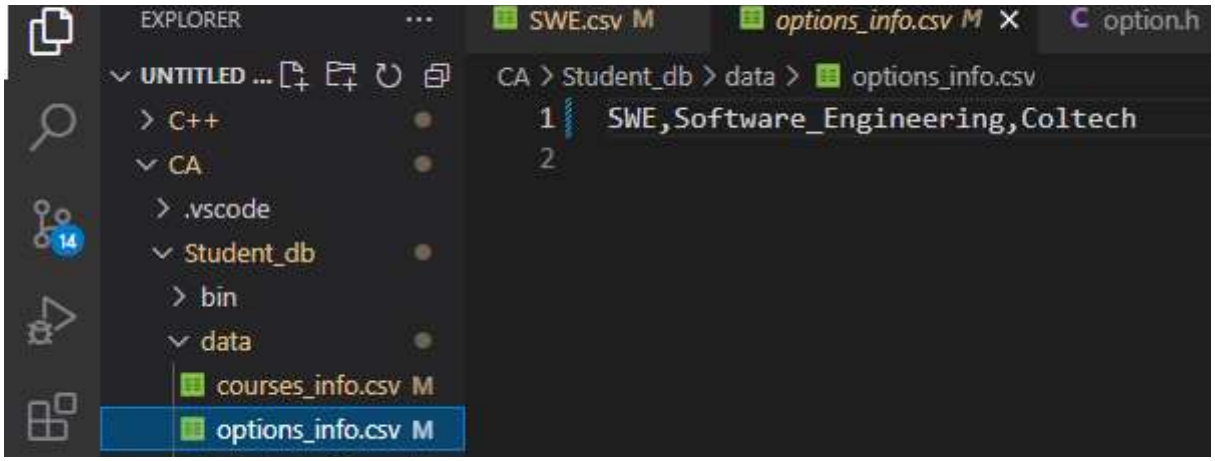
```
"D:\My Files\Documents\School-Work\C++\CA\Student_db\main.exe"
How many courses do you want to register: 5
Enter course code: cen1
Enter course name: algorithm
Enter course credit value: 3
algorithm Successfully Added
Enter course code: cen2
Enter course name: object-oriented
Enter course credit value: 4
object-oriented Successfully Added
Enter course code: cen3
Enter course name: python
Enter course credit value: 2
python Successfully Added
Enter course code: data_structures
Enter course name: data_structures
Enter course credit value: 2
data_structures Successfully Added
Enter course code: cen5
Enter course name: networks
Enter course credit value: 4
networks Successfully Added

Process returned 0 (0x0)   execution time : 96.407 s
Press any key to continue.
```



- **Creating Department**

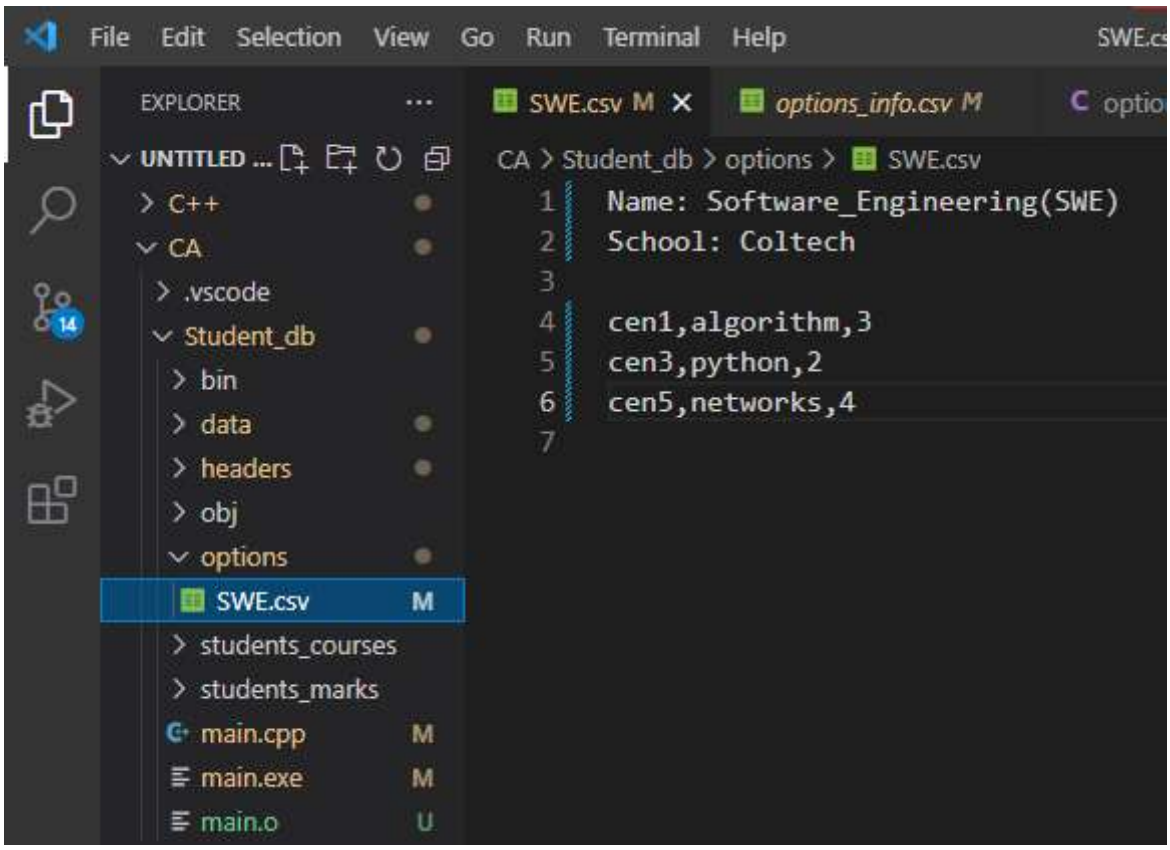
- 





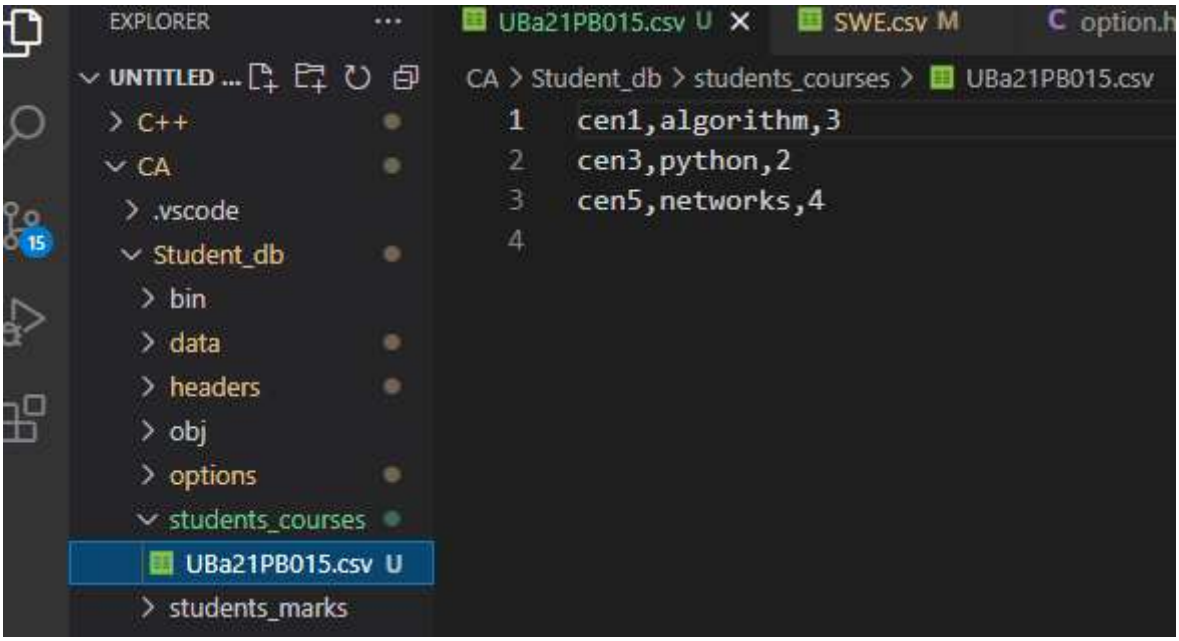
- **Adding departmental courses**

```
1: cen1-algorithm-3
2: cen2-object-oriented-4
3: cen3-python-2
4: data_structures-data_structures-2
5: cen5-networks-4
Enter the number(s) of the courses you want to register
Seperated them with a spaces
Hit enter when you are done: 1 3 5
added courses
Process returned 0 (0x0)   execution time : 194.470 s
Press any key to continue.
```



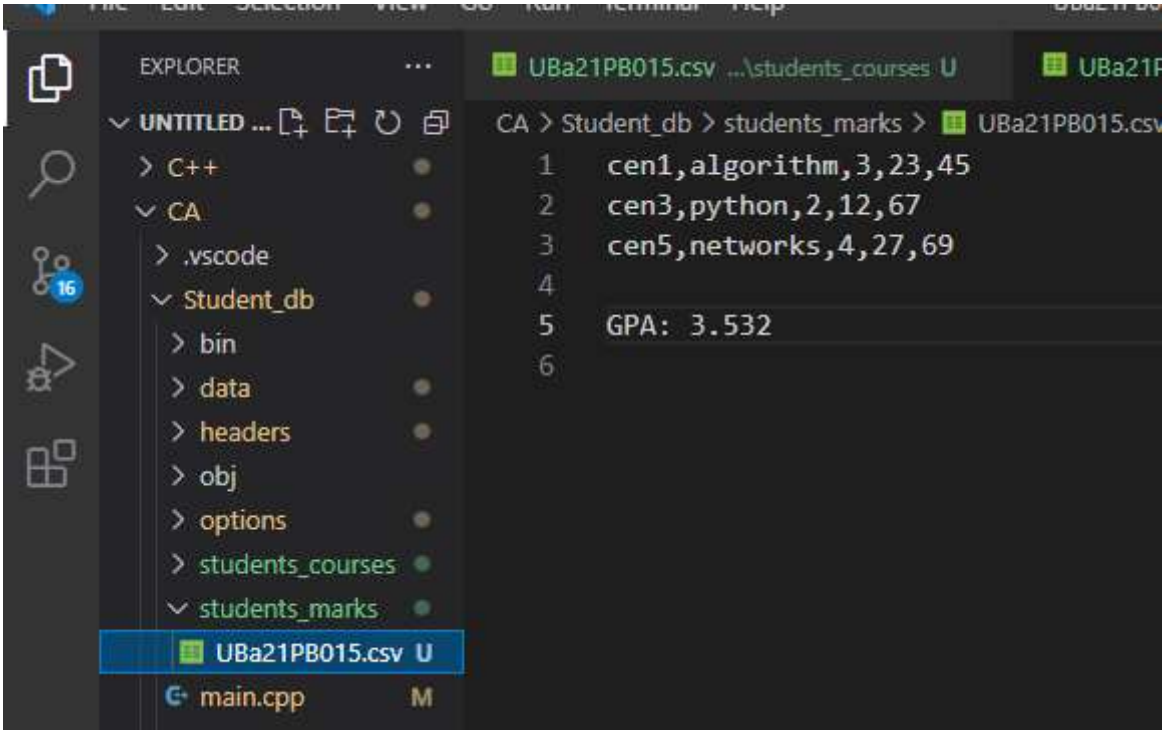
- **Registering student courses from departmental courses**

```
registered courses

Process returned 0 (0x0)   execution time : 0.067 s
Press any key to continue.
```



- **Registering Students Marks**

```
Enter marks for algorithm,
CA Marks: 23
Exam Mark: 45
Enter marks for python,
CA Marks: 12
Exam Mark: 67
Enter marks for networks,
CA Marks: 27
Exam Mark: 69
Marks uploaded

Process returned 0 (0x0)   execution time : 23.931 s
Press any key to continue.
```


EXPLORER
UNTITLED ...
C++
CA
.vscode
Student_db
bin
data
headers
obj
options
students_courses
students_marks
UBa21PB015.csv U
main.cpp M
UBa21PB015.csv ...\students_courses U
CA > Student_db > students_marks > UBa21PB015.csv
cen1,algorithm,3,23,45
cen3,python,2,12,67
cen5,networks,4,27,69
GPA: 3.532

Assignments

1. User ID’s

Many Organizations have user ids which are constrained in some way. Imagine you work at an internet service provider and the user ids are all two letters followed by two numbers (e.g. aa49). Your task at such an organization might be to hold a record ont the billing activity for each possible user.

Task: Write a function which creates a list of all possible user ids. Assume the letters are all lower case.

Input:
Lowercase = ‘abcdefghijklmnopqrstuvwxyz’
Digits = ‘0123456789’

Function:

ids.cpp > ...

```cpp
#include <iostream>
#include <string>

using namespace std;

// function to generate ids
string generateID(string letters, string digits){
    int i, j, k, l;
    string all_ids;

    // first digit
    for (i = 0; i < 10; i++)
        // second digit
        for (j = 0; j < 10; j++)
            // First letter
            for (k = 0; k < letters.length(); k++)
                // Second letter
                for (l = 0; l < letters.length(); l++)
                    all_ids += to_string(i) + to_string(j) + letters[k] + letters[l] + ",";
    return all_ids;
}
```

Main Code:

```cpp
int main(){
    string possible_ids = generateID("abcdefghijklmnopqrstuvwxyz", "0123456789");
    int i, count = 0;
    for (i = 0; i < possible_ids.length(); i++){
        if (possible_ids[i+4] == ','){
                cout << "\t";
                count++;
        }
        cout << possible_ids[i];
    }

    cout << "\n------------------------- \nThere are " << count << " possible ids" << endl;
    return 0;
}
```

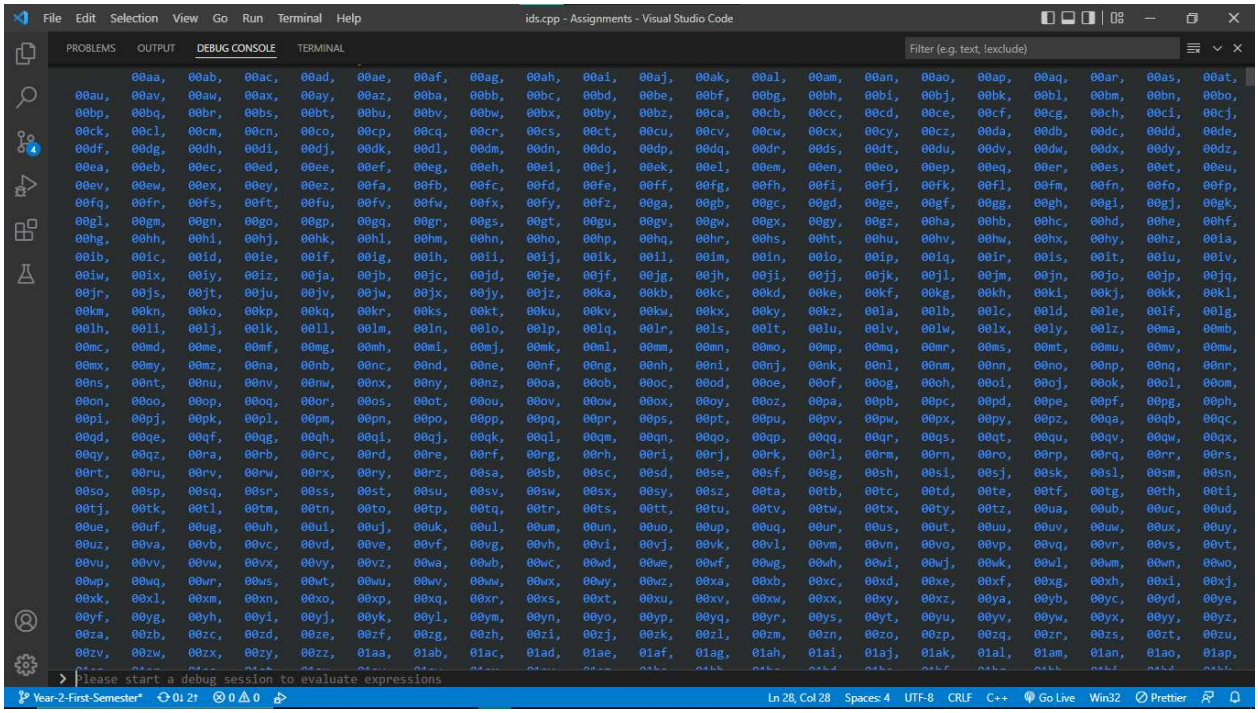
Function (using recursion):

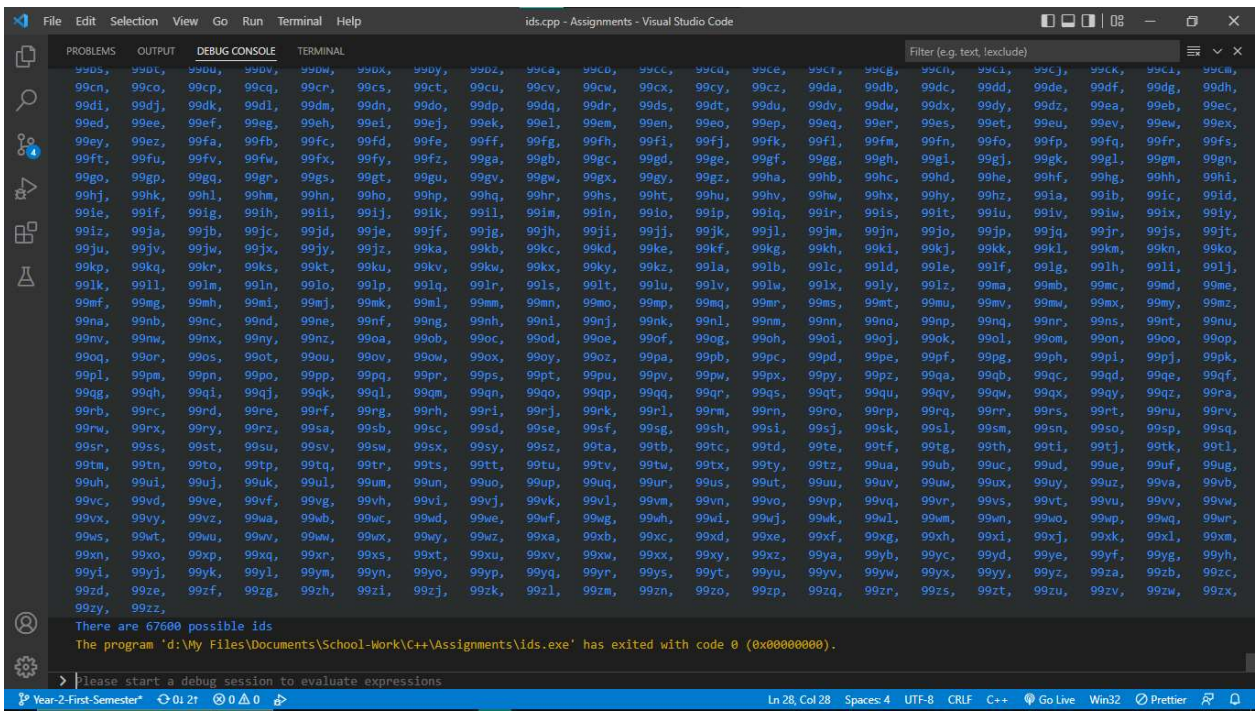
ids_with_recursion.cpp > ...

```cpp
#include <iostream>
#include <string>
using namespace std;

string letters = "abcdefghijklmnopqrstuvwxyz";
int count = 0;

// id function
string genIDs(int pos[], bool checkend){

    string ids = to_string(pos[0]) + to_string(pos[1]) + letters[pos[2]] + letters[pos[3]] + ",  ";

    pos[3]++;
    if (pos[3] == 26){
        pos[2]++;
        pos[3] = 0;
    }

    if (pos[2] == 26){
        pos[1]++;
        pos[2] = 0;
        pos[3] = 0;
    }

    if (pos[1] == 10){
        pos[0]++;
        pos[1] = 0;
        pos[2] = 0;
        pos[3] = 0;
    }

    count++;

    if (pos[0] == 10){
        return ids;
    }

    cout << ids <<"\t";
    return ids + genIDs(pos, checkend);
};
```

```cpp

int main() {
    int positions[] = {0,0,0,0};
    string ids = genIDs(positions, false);
    cout << ids << endl << "--------------------\n";
    cout << "there are " << count << " number of ids";
    return 0;
}
```

Output:


File Edit Selection View Go Run Terminal Help
ids.cpp - Assignments - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Filter (e.g. text, !exclude)
00aa, 00ab, 00ac, 00ad, 00ae, 00af, 00ag, 00ah, 00ai, 00aj, 00ak, 00al, 00am, 00an, 00ao, 00ap, 00aq, 00ar, 00as, 00at,
00au, 00av, 00aw, 00ax, 00ay, 00az, 00ba, 00bb, 00bc, 00bd, 00be, 00bf, 00bg, 00bh, 00bi, 00bj, 00bk, 00bl, 00bm, 00bn, 00bo,
00bp, 00bq, 00br, 00bs, 00bt, 00bu, 00bv, 00bw, 00bx, 00by, 00bz, 00ca, 00cb, 00cc, 00cd, 00ce, 00cf, 00cg, 00ch, 00ci, 00cj,
00ck, 00cl, 00cm, 00cn, 00co, 00cp, 00cq, 00cr, 00cs, 00ct, 00cu, 00cv, 00cw, 00cx, 00cy, 00cz, 00da, 00db, 00dc, 00dd, 00de,
00df, 00dg, 00dh, 00di, 00dj, 00dk, 00dl, 00dm, 00dn, 00do, 00dp, 00dq, 00dr, 00ds, 00dt, 00du, 00dv, 00dw, 00dx, 00dy, 00dz,
00ea, 00eb, 00ec, 00ed, 00ee, 00ef, 00eg, 00eh, 00ei, 00ej, 00ek, 00el, 00em, 00en, 00eo, 00ep, 00eq, 00er, 00es, 00et, 00eu,
00ev, 00ew, 00ex, 00ey, 00ez, 00fa, 00fb, 00fc, 00fd, 00fe, 00ff, 00fg, 00fh, 00fi, 00fj, 00fk, 00fl, 00fm, 00fn, 00fo, 00fp,
00fq, 00fr, 00fs, 00ft, 00fu, 00fv, 00fw, 00fx, 00fy, 00fz, 00ga, 00gb, 00gc, 00gd, 00ge, 00gf, 00gg, 00gh, 00gi, 00gj, 00gk,
00gl, 00gm, 00gn, 00go, 00gp, 00gq, 00gr, 00gs, 00gt, 00gu, 00gv, 00gw, 00gx, 00gy, 00gz, 00ha, 00hb, 00hc, 00hd, 00he, 00hf,
00hg, 00hh, 00hi, 00hj, 00hk, 00hl, 00hm, 00hn, 00ho, 00hp, 00hq, 00hr, 00hs, 00ht, 00hu, 00hv, 00hw, 00hx, 00hy, 00hz, 00ia,
00ib, 00ic, 00id, 00ie, 00if, 00ig, 00ih, 00ii, 00ij, 00ik, 00il, 00im, 00in, 00io, 00ip, 00iq, 00ir, 00is, 00it, 00iu, 00iv,
00iw, 00ix, 00iy, 00iz, 00ja, 00jb, 00jc, 00jd, 00je, 00jf, 00jg, 00jh, 00ji, 00jj, 00jk, 00jl, 00jm, 00jn, 00jo, 00jp, 00jq,
00jr, 00js, 00jt, 00ju, 00jv, 00jw, 00jx, 00jy, 00jz, 00ka, 00kb, 00kc, 00kd, 00ke, 00kf, 00kg, 00kh, 00ki, 00kj, 00kk, 00kl,
00km, 00kn, 00ko, 00kp, 00kq, 00kr, 00ks, 00kt, 00ku, 00kv, 00kw, 00kx, 00ky, 00kz, 00la, 00lb, 00lc, 00ld, 00le, 00lf, 00lg,
00lh, 00li, 00lj, 00lk, 00ll, 00lm, 00ln, 00lo, 00lp, 00lq, 00lr, 00ls, 00lt, 00lu, 00lv, 00lw, 00lx, 00ly, 00lz, 00ma, 00mb,
00mc, 00md, 00me, 00mf, 00mg, 00mh, 00mi, 00mj, 00mk, 00ml, 00mm, 00mn, 00mo, 00mp, 00mq, 00mr, 00ms, 00mt, 00mu, 00mv, 00mw,
00mx, 00my, 00mz, 00na, 00nb, 00nc, 00nd, 00ne, 00nf, 00ng, 00nh, 00ni, 00nj, 00nk, 00nl, 00nm, 00nn, 00no, 00np, 00nq, 00nr,
00ns, 00nt, 00nu, 00nv, 00nw, 00nx, 00ny, 00nz, 00oa, 00ob, 00oc, 00od, 00oe, 00of, 00og, 00oh, 00oi, 00oj, 00ok, 00ol, 00om,
00on, 00oo, 00op, 00oq, 00or, 00os, 00ot, 00ou, 00ov, 00ow, 00ox, 00oy, 00oz, 00pa, 00pb, 00pc, 00pd, 00pe, 00pf, 00pg, 00ph,
00pi, 00pj, 00pk, 00pl, 00pm, 00pn, 00po, 00pp, 00pq, 00pr, 00ps, 00pt, 00pu, 00pv, 00pw, 00px, 00py, 00pz, 00qa, 00qb, 00qc,
00qd, 00qe, 00qf, 00qg, 00qh, 00qi, 00qj, 00qk, 00ql, 00qm, 00qn, 00qo, 00qp, 00qq, 00qr, 00qs, 00qt, 00qu, 00qv, 00qw, 00qx,
00qy, 00qz, 00ra, 00rb, 00rc, 00rd, 00re, 00rf, 00rg, 00rh, 00ri, 00rj, 00rk, 00rl, 00rm, 00rn, 00ro, 00rp, 00rq, 00rr, 00rs,
00rt, 00ru, 00rv, 00rw, 00rx, 00ry, 00rz, 00sa, 00sb, 00sc, 00sd, 00se, 00sf, 00sg, 00sh, 00si, 00sj, 00sk, 00sl, 00sm, 00sn,
00so, 00sp, 00sq, 00sr, 00ss, 00st, 00su, 00sv, 00sw, 00sx, 00sy, 00sz, 00ta, 00tb, 00tc, 00td, 00te, 00tf, 00tg, 00th, 00ti,
00tj, 00tk, 00tl, 00tm, 00tn, 00to, 00tp, 00tq, 00tr, 00ts, 00tt, 00tu, 00tv, 00tw, 00tx, 00ty, 00tz, 00ua, 00ub, 00uc, 00ud,
00ue, 00uf, 00ug, 00uh, 00ui, 00uj, 00uk, 00ul, 00um, 00un, 00uo, 00up, 00uq, 00ur, 00us, 00ut, 00uu, 00uv, 00uw, 00ux, 00uy,
00uz, 00va, 00vb, 00vc, 00vd, 00ve, 00vf, 00vg, 00vh, 00vi, 00vj, 00vk, 00vl, 00vm, 00vn, 00vo, 00vp, 00vq, 00vr, 00vs, 00vt,
00vu, 00vv, 00vw, 00vx, 00vy, 00vz, 00wa, 00wb, 00wc, 00wd, 00we, 00wf, 00wg, 00wh, 00wi, 00wj, 00wk, 00wl, 00wm, 00wn, 00wo,
00wp, 00wq, 00wr, 00ws, 00wt, 00wu, 00wv, 00ww, 00wx, 00wy, 00wz, 00xa, 00xb, 00xc, 00xd, 00xe, 00xf, 00xg, 00xh, 00xi, 00xj,
00xk, 00xl, 00xm, 00xn, 00xo, 00xp, 00xq, 00xr, 00xs, 00xt, 00xu, 00xv, 00xw, 00xx, 00xy, 00xz, 00ya, 00yb, 00yc, 00yd, 00ye,
00yf, 00yg, 00yh, 00yi, 00yj, 00yk, 00yl, 00ym, 00yn, 00yo, 00yp, 00yq, 00yr, 00ys, 00yt, 00yu, 00yv, 00yw, 00yx, 00yy, 00yz,
00za, 00zb, 00zc, 00zd, 00ze, 00zf, 00zg, 00zh, 00zi, 00zj, 00zk, 00zl, 00zm, 00zn, 00zo, 00zp, 00zq, 00zr, 00zs, 00zt, 00zu,
00zv, 00zw, 00zx, 00zy, 00zz, 01aa, 01ab, 01ac, 01ad, 01ae, 01af, 01ag, 01ah, 01ai, 01aj, 01ak, 01al, 01am, 01an, 01ao, 01ap,
Please start a debug session to evaluate expressions
Year-2-First-Semester*
Ln 28, Col 28 Spaces: 4 UTF-8 CRLF C++ Go Live Win32 Prettier


File Edit Selection View Go Run Terminal Help
ids.cpp - Assignments - Visual Studio Code
PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL
Filter (e.g. text, !exclude)
99cn, 99co, 99cp, 99cq, 99cr, 99cs, 99ct, 99cu, 99cv, 99cw, 99cx, 99cy, 99cz, 99da, 99db, 99dc, 99dd, 99de, 99df, 99dg, 99dh,
99di, 99dj, 99dk, 99dl, 99dm, 99dn, 99do, 99dp, 99dq, 99dr, 99ds, 99dt, 99du, 99dv, 99dw, 99dx, 99dy, 99dz, 99ea, 99eb, 99ec,
99ed, 99ee, 99ef, 99eg, 99eh, 99ei, 99ej, 99ek, 99el, 99em, 99en, 99eo, 99ep, 99eq, 99er, 99es, 99et, 99ev, 99ew, 99ex,
99ey, 99ez, 99fa, 99fb, 99fc, 99fd, 99fe, 99ff, 99fg, 99fh, 99fi, 99fj, 99fk, 99fl, 99fm, 99fn, 99fo, 99fp, 99fq, 99fr, 99fs,
99ft, 99fu, 99fv, 99fw, 99fx, 99fy, 99fz, 99ga, 99gb, 99gc, 99gd, 99ge, 99gf, 99gg, 99gh, 99gi, 99gj, 99gk, 99gl, 99gm, 99gn,
99go, 99gp, 99gq, 99gr, 99gs, 99gt, 99gu, 99gv, 99gw, 99gx, 99gy, 99gz, 99ha, 99hb, 99hc, 99hd, 99he, 99hf, 99hg, 99hh, 99hi,
99hj, 99hk, 99hl, 99hm, 99hn, 99ho, 99hp, 99hq, 99hr, 99hs, 99ht, 99hu, 99hv, 99hw, 99hx, 99hy, 99hz, 99ia, 99ib, 99ic, 99id,
99ie, 99if, 99ig, 99ih, 99ii, 99ij, 99ik, 99il, 99im, 99in, 99io, 99ip, 99iq, 99ir, 99is, 99it, 99iu, 99iv, 99iw, 99ix, 99iy,
99iz, 99ja, 99jb, 99jc, 99jd, 99je, 99jf, 99jg, 99jh, 99ji, 99jj, 99jk, 99jl, 99jm, 99jn, 99jo, 99jp, 99jq, 99jr, 99js, 99jt,
99ju, 99jv, 99jw, 99jx, 99jy, 99jz, 99ka, 99kb, 99kc, 99kd, 99ke, 99kf, 99kg, 99kh, 99ki, 99kj, 99kk, 99kl, 99km, 99kn, 99ko,
99kp, 99kq, 99kr, 99ks, 99kt, 99ku, 99kv, 99kw, 99kx, 99ky, 99kz, 99la, 99lb, 99lc, 99ld, 99le, 99lf, 99lg, 99lh, 99li, 99lj,
99lk, 99ll, 99lm, 99ln, 99lo, 99lp, 99lq, 99lr, 99ls, 99lt, 99lu, 99lv, 99lw, 99lx, 99ly, 99lz, 99ma, 99mb, 99mc, 99md, 99me,
99mf, 99mg, 99mh, 99mi, 99mj, 99mk, 99ml, 99mm, 99mn, 99mo, 99mp, 99mq, 99mr, 99ms, 99mt, 99mu, 99mv, 99mw, 99mx, 99my, 99mz,
99na, 99nb, 99nc, 99nd, 99ne, 99nf, 99ng, 99nh, 99ni, 99nj, 99nk, 99nl, 99nm, 99nn, 99no, 99np, 99nq, 99nr, 99ns, 99nt, 99nu,
99nv, 99nw, 99nx, 99ny, 99nz, 99oa, 99ob, 99oc, 99od, 99oe, 99of, 99og, 99oh, 99oi, 99oj, 99ok, 99ol, 99om, 99on, 99oo, 99op,
99oq, 99or, 99os, 99ot, 99ou, 99ov, 99ow, 99ox, 99oy, 99oz, 99pa, 99pb, 99pc, 99pd, 99pe, 99pf, 99pg, 99ph, 99pi, 99pj, 99pk,
99pl, 99pm, 99pn, 99po, 99pp, 99pq, 99pr, 99ps, 99pt, 99pu, 99pv, 99pw, 99px, 99py, 99pz, 99qa, 99qb, 99qc, 99qd, 99qe, 99qf,
99qg, 99qh, 99qi, 99qj, 99qk, 99ql, 99qm, 99qn, 99qo, 99qp, 99qq, 99qr, 99qs, 99qt, 99qu, 99qv, 99qw, 99qx, 99qy, 99qz, 99ra,
99rb, 99rc, 99rd, 99re, 99rf, 99rg, 99rh, 99ri, 99rj, 99rk, 99rl, 99rm, 99rn, 99ro, 99rp, 99rq, 99rr, 99rs, 99rt, 99ru, 99rv,
99rw, 99rx, 99ry, 99rz, 99sa, 99sb, 99sc, 99sd, 99se, 99sf, 99sg, 99sh, 99si, 99sj, 99sk, 99sl, 99sm, 99sn, 99so, 99sp, 99sq,
99sr, 99ss, 99st, 99su, 99sv, 99sw, 99sx, 99sy, 99sz, 99ta, 99tb, 99tc, 99td, 99te, 99tf, 99tg, 99th, 99ti, 99tj, 99tk, 99tl,
99tm, 99tn, 99to, 99tp, 99tq, 99tr, 99ts, 99tt, 99tu, 99tv, 99tw, 99tx, 99ty, 99tz, 99ua, 99ub, 99uc, 99ud, 99ue, 99uf, 99ug,
99uh, 99ui, 99uj, 99uk, 99ul, 99um, 99un, 99uo, 99up, 99uq, 99ur, 99us, 99ut, 99uu, 99uv, 99uw, 99ux, 99uy, 99uz, 99va, 99vb,
99vc, 99vd, 99ve, 99vf, 99vg, 99vh, 99vi, 99vj, 99vk, 99vl, 99vm, 99vn, 99vo, 99vp, 99vq, 99vr, 99vs, 99vt, 99vu, 99vv, 99vw,
99vx, 99vy, 99vz, 99wa, 99wb, 99wc, 99wd, 99we, 99wf, 99wg, 99wh, 99wi, 99wj, 99wk, 99wl, 99wm, 99wn, 99wo, 99wp, 99wq, 99wr,
99ws, 99wt, 99wu, 99wv, 99ww, 99wx, 99wy, 99wz, 99xa, 99xb, 99xc, 99xd, 99xe, 99xf, 99xg, 99xh, 99xi, 99xj, 99xk, 99xl, 99xm,
99xn, 99xo, 99xp, 99xq, 99xr, 99xs, 99xt, 99xu, 99xv, 99xw, 99xx, 99xy, 99xz, 99ya, 99yb, 99yc, 99yd, 99ye, 99yf, 99yg, 99yh,
99yi, 99yj, 99yk, 99yl, 99ym, 99yn, 99yo, 99yp, 99yq, 99yr, 99ys, 99yt, 99yu, 99yv, 99yw, 99yx, 99yy, 99yz, 99za, 99zb, 99zc,
99zd, 99ze, 99zf, 99zg, 99zh, 99zi, 99zj, 99zk, 99zl, 99zm, 99zn, 99zo, 99zp, 99zq, 99zr, 99zs, 99zt, 99zu, 99zv, 99zw, 99zx,
99zy, 99zz,
There are 67600 possible ids
The program 'd:\My Files\Documents\School-Work\C++\Assignments\ids.exe' has exited with code 0 (0x00000000).
Please start a debug session to evaluate expressions
Year-2-First-Semester*
Ln 28, Col 28 Spaces: 4 UTF-8 CRLF C++ Go Live Win32 Prettier

2. Triangles

a. Pyramid

triangles.cpp > pyramid(int)

```cpp
#include <iostream>
using namespace std;

void pyramid(int height){
    int i, j, k;

    cout << "PYRAMID\n----------------\n\t ";

    for (i = 0; i < height; i++)
        cout << i+1 << " ";

    cout << "\n\n";

    for (i = 0; i < height; i++){
        cout << i+1 << "\t";
        for (k = height - i - 1; k > 0; k--)
            cout << " ";
        for (j = 0; j <= i; j++)
            cout << "* ";
        cout << endl;
    }
    cout << "\n\n";
}
```

b. Right Angle

```cpp
void rightAngle(int height){
    int i, j;

    cout << "RIGHT ANGLE\n----------------\n\t ";

    for (i = 0; i < height; i++)
        cout << i+1 << " ";

    cout << "\n\n";

    for (i = 0; i < height; i++){
        cout << i+1 << "\t";
        for (j = 0; j <= i; j++)
            cout << " *";
        cout << endl;
    }
    cout << "\n\n";
}
```

c. Half diamond

```
void halfDiamond(int radius){
    int i, j, k;

    cout << "HALF DIAMOND\n----------------\n\t ";

    for (i = 0; i < radius; i++)
        cout << i+1 << " ";

    cout << "\n\n";

    for (i = 0; i < radius; i++){
        cout << i+1 << "\t";
        for(k =  radius - i - 1; k > 0; k--)
            cout << "  ";

        for(j = 0; j <= i; j++)
            cout << "* ";
        cout << endl;
    }
    int count = i;
    for (i = radius - 2; i >= 0; i--){
        cout << ++count << "\t";
        for(k = 1; k < radius - i; k++)
            cout << "  ";

        for(j = 0; j <= i; j++)
            cout << "* ";
        cout << endl;
    }
    cout << "\n\n";
}
```

Main Code

```
int main(){
    int height;

    do {
        cout << "Enter height ( > 0): ";
        cin >> height;
    } while(height < 1);

    pyramid(height);
    rightAngle(height);
    halfDiamond(height/2);
}
```

Output:

```
File  Edit  Selection  View  Go  Run  Terminal  Help

PROBLEMS   OUTPUT   DEBUG CONSOLE   TERMINAL

Enter height ( > 0 ): 9
PYRAMID
        1 2 3 4 5 6 7 8 9
1             *
2            * *
3           * * *
4          * * * *
5         * * * * *
6        * * * * * *
7       * * * * * * *
8      * * * * * * * *
9     * * * * * * * * *
RIGHT ANGLE
        1 2 3 4 5 6 7 8 9
1       *
2       * *
3       * * *
4       * * * *
5       * * * * *
6       * * * * * *
7       * * * * * * *
8       * * * * * * * *
9       * * * * * * * * *
HALF DIAMOND
        1 2 3 4
1           *
2         * *
3       * * *
4     * * * *
5       * * *
6         * *
7           *
The program 'd:\My Files\Documents\School-Work\C++\As
```

3. Factorial of a number

Function (iteration):

factorial.cpp > main()

```cpp
#include <iostream>

using namespace std;

// factorial function
int factorial(int num){
    int fact = 1, i;
    for (i = num; i > 0; i--){
        fact *= i;
    }
    return fact;
}

int main(){
    int num;
    do {
        cout << "Enter a positive number: ";
        cin >> num;
    } while (num < 0);
    cout << "The factorial of " << num << " is " << factorial(num);
    return 0;
}
```

Function (resursive):

factorial_recur.cpp > main()

```cpp
#include <iostream>

using namespace std;

// factorial function
int factorial(int num){
    if (num == 1 || num == 0){
        return 1;
    }
    return num * factorial(num - 1);
}

int main(){
    int num ;
    do {
        cout << "Enter a positive number: ";
        cin >> num;
    } while (num < 0);
    cout << "The factorial of " << num << " is " << factorial(num);
    return 0;
}
```

Output:

```
[Running] cd "d:\My Files\Documents\School-Work\C++\
The factorial of 6 is 720
```

4. Random Guessing Game

Function returns true if user wins else false.
Function takes as parameter, the range of random values and the number of chances

guessing_game.cpp > playGuessingGame(int, int, int)

```cpp
#include <iostream>
#include <cstdlib>

using namespace std;

bool playGuessingGame(int lowest_number, int highest_number, int chances){

    // generates a random game between between
    // lowest_number and highest_number inclusively
    int guess = rand() % (highest_number - lowest_number + 1) + lowest_number;
    int num;

    while(chances != 0) {
        cout << "Enter a number (" << chances << " chances left): ";
        cin >> num;

        // returns true if number matches
        if (num == guess){
            return true;
        }

        else if (guess > num) {
            cout << "The answer is greater than " << num << endl;
        }

        else if (guess < num) {
            cout << "The answer is less than " << num << endl;
        }
        chances--;
    }
    cout << "Answer is " << guess << endl;
    return false;
}

int main(){
    int i = 0;
    bool hasWon = playGuessingGame(1, 100, 8);
    // similar to if...else
    hasWon ? cout << "You Have Won The Game" : cout << "You Have Lossed The Game";
    return 0;
}
```

Output (loss):

```
[New Thread 11156.0x2a24]
Enter a number (8 chances left): 98
The answer is less than 98
Enter a number (7 chances left): 12
The answer is greater than 12
Enter a number (6 chances left): 23
The answer is greater than 23
Enter a number (5 chances left): 34
The answer is greater than 34
Enter a number (4 chances left): 44
The answer is less than 44
Enter a number (3 chances left): 57
The answer is less than 57
Enter a number (2 chances left): 71
The answer is less than 71
Enter a number (1 chances left): 85
The answer is less than 85
Answer is 42
You Have Lossed The Game
```

Output (win):

```
Enter a number (8 chances left): 98
The answer is less than 98
Enter a number (7 chances left): 12
The answer is greater than 12
Enter a number (6 chances left): 23
The answer is greater than 23
Enter a number (5 chances left): 71
The answer is less than 71
Enter a number (4 chances left): 63
You Have Won The Game
```

5. R = V/I

Code:

resistance.cpp > main()

```cpp
#include <iostream>

using namespace std;

// function to get a positive floating number
float getPositiveFloat(){
    float value;
    do {
        cin >> value;
    } while(value < 0);
    return value;
}

int main(){
    float R, V, I;

    cout << "Enter Voltage (V): ";
    V = getPositiveFloat();

    cout << "Enter Current (I): ";
    I = getPositiveFloat();

    R = V/I;

    cout << "R = " << R << endl;
}
```

Output:

```
Enter Voltage (V): 220
Enter Current (I): 5
R = 44
```

6. Array Search

Code:

array_search.cpp > main()

```cpp

int searchArray(int array[], int length, int value){
    int i;

    for (i = 0; i < length; i++){
        if(value == array[i])
            return i+1;
    }
    return -1;
}

int main(){
    int myArray[] = {23, 67, 0, 9, 6, 102, 17};
    int value;
    cout << "Enter a value: ";
    cin >> value;
    int pos = searchArray(myArray, 7, value);
    (pos == -1) ?  cout << "Not Found\n" : cout << "Found at " << pos << endl;
    return 0;
}
```

Outputs:

```
Enter a value: 10
Not Found
```

```
Enter a value: 102
Found at 6
```

7. Upper Case

Code:

upperCase.cpp M ×

upperCase.cpp > upperCase(string)

```cpp
#include <iostream>

using namespace std;

string upperCase(string text){

    // capitallise the whole text
    int i;
    for(i = 0; i < text.length(); i++){
        text[i] = toupper(text[i]);
    }

    return text;
}

string sentenceCase(string text){

    // capitallise the first letter
    text[0] = toupper(text[0]);

    // capitallise the first letter of each word
    int i;
    for(i = 0; i < text.length() - 1; i++){
        if(text[i] == ' '){
            text[i+1] = toupper(text[i+1]);
        }
    }

    return text;
}


int main(){
    string sentence = "my name is mbah lesky";
    string name = "mbah lesky";

    cout << "\"" << name << "\"" << " in upper case is " << "\"" << upperCase(name) << "\"\n";
    cout << "\"" << sentence << "\"" << " in sentence case is " << "\"" << sentenceCase(sentence) << "\"\n";
    return 0;
}
```

Output:

```
[Running] cd "d:\My Files\Documents\School-Work\C++\Assignments\" && g+
"mbah lesky" in upper case is "MBAH LESKY"
"my name is mbah lesky" in sentence case is "My Name Is Mbah Lesky"
```

8. Operator Overloading

Class:

string_operator_overload.cpp > main()

```cpp
#include <iostream>

using namespace std;

class MyString {
    public:
    string value;

    // empty constructor
    MyString(){}

    // + operator overload
    // use to add two objects together to produce one
    MyString operator+(MyString another){
        MyString result;
        result.value = this->value + another.value;
        return result;
    }

};

int main(){
    MyString firstname;
    MyString secondname;
    firstname.value = "Mbah";
    secondname.value = "Lesky";
    MyString fullname = firstname + secondname;

 //   MyString fullname = firstname + secondname;
    cout << firstname.value << " + " << secondname.value << " = " << fullname.value << endl;
    return 0;
}
```

Output:

```
Mbah + Lesky = MbahLesky
```