



# C++ PROGRAMMING (335)

## REGIONAL – 2019

**Production Portion:**

Job 1: Department Graph \_\_\_\_\_ (320 points)

***TOTAL POINTS*** \_\_\_\_\_ (320 points)

**Graders: Please double check and verify all scores  
and answer keys!**

Property of Business Professionals of America.  
May be reproduced only for use in the Business Professionals of America  
*Workplace Skills Assessment Program* competition.



Your application will be graded on the following criteria:

**Solution and Project**

Project was found on the flash drive \_\_\_\_\_ 10 points  
Project is named following the naming convention \_\_\_\_\_ 10 points

**Program Execution**

Program runs \_\_\_\_\_ 20 points

**If program does not execute, then remaining items in this section are not scored.**

The program gracefully handles any data errors \_\_\_\_\_ 20 points

The program shows a department menu to choose from \_\_\_\_\_ 10 points

The program displays a prompt for collecting student sizes from the user  
which includes which class is currently be collected. \_\_\_\_\_ 10 points

The program displays an error message for bad user data \_\_\_\_\_ 10 points

The program displays both types of graphs \_\_\_\_\_ 20 points

The program displays prompts for user control \_\_\_\_\_ 20 points

The program shows all data on clean console screens \_\_\_\_\_ 10 points

The program allows the user to rerun the program without closing  
and restarting the current copy. \_\_\_\_\_ 5 points

**Source Code Review**

Contestant ID is commented at the top of the program \_\_\_\_\_ 5 points

Code is commented at the top, for each method and as needed \_\_\_\_\_ 15 points

Code uses reasonable and consistent variable naming conventions \_\_\_\_\_ 15 points

The program reads the class sizes into the correct data structure \_\_\_\_\_ 10 points

The program gracefully handles remaining error checking \_\_\_\_\_ 20 points

A method called "menu\_collect" is implemented \_\_\_\_\_ 30 points

A method called "horizontal\_graph" is implemented \_\_\_\_\_ 30 points

A method called "vertical\_graph" is implemented \_\_\_\_\_ 40 points

The program uses the bool data type in a logical manor \_\_\_\_\_ 10 points

**Total** \_\_\_\_\_/320 points



```
// Golden Oaks.cpp : Defines the entry point for the console application.
//
```

```
#include "stdafx.h"
#include <iostream>
#include <iomanip>
#include <cstdlib>
#include <ctime> // needed for rand()% seed
#include <string>
#include <vector>
#include <fstream>
#include <conio.h>
```

```
using namespace std;
```

```
char Ans;
```

```
vector <vector <char>> V_Graph(10);
```

```
// size of vector V_Graph.size() and V_Graph[0].size()
```

```
vector<int> Classes(10);
```

```
int I, J;
```

```
int Dept;
```

```
void menu_collect(int &, vector <int> &);
```

```
void horizontal_graph(int , vector <int> , vector <vector <char>>);
```

```
void vertical_graph(int, vector <int>, vector <vector <char>>);
```

```
void main()
```

```
{
    do {
        system("CLS");
        cout << setiosflags(ios::right);
        // Init V_Graph to empty
        for (I = 0; I < V_Graph.size(); I++)
        {
            V_Graph[I].resize(40);
            for (J = 0; J < V_Graph[0].size(); J++)
                V_Graph[I][J] = ' ';
        }
        menu_collect(Dept, Classes);
        horizontal_graph(Dept, Classes, V_Graph);
        vertical_graph(Dept, Classes, V_Graph);

        cout << "Run the program again (Y/y)?";
        cin >> Ans;
    } while (Ans == 'Y' || Ans == 'y');
}
```



```

void menu_collect(int & Dept, vector<int> &Classes)
{
    int I;
    string Trash;

    // Collect the Department
    do {
        system("CLS");
        cout << setw(40) << "Up Town University" << endl << endl;
        cout << setw(20) << "0." << " English Department" << endl;
        cout << setw(20) << "1." << " Mathematics Department" << endl;
        cout << setw(20) << "2." << " Computer Science Department" << endl;
        cout << setw(20) << "3." << " Business Department" << endl;
        cout << setw(20) << "4." << " Kinesiology Department" << endl;
        cout << setw(20) << "5." << " Architecture Department" << endl;
        cout << setw(20) << "6." << " Biology Department" << endl;
        cout << setw(20) << "7." << " Education Department" << endl;
        cout << setw(20) << "8." << " Chemistry Department" << endl;
        cout << setw(20) << "9." << " Engineering Department" << endl<<endl;
        cout << setw(45) << "Please pick the department:";
        cin >> Dept;

        // Clean the buffer so getchar works.
        getline(cin, Trash);
        if (Dept < 0 || Dept > 9)
        {
            cout<<endl <<setw(50)<< "Please enter only the number 0 -9." << endl;
            cout << setw(45) << "Press ENTER to continue." << endl;
            // hold message on the screen until the user presses Enter
            getchar();
        }
    } while (Dept < 0 || Dept > 9);

    // Collect the class sizes
    system("CLS");
    for (I = 0; I < V_Graph.size(); I++)
    {
        do
        {
            cout << "Class sizes are from 0 - 40 students." << endl;
            cout << Dept << "0" << I<<" ";
            cin >> Classes[I];
            if (Classes[I] < 0 || Classes[I] > 40)
                cout << "Please enter ONLY numbers 0 - 40" << endl<<endl;
        } while (Classes[I] < 0 || Classes[I] > 40);
    }

    // Fill V_Graph with students using *
    for (I = 0; I < V_Graph.size(); I++)
        for (J = 0; J < Classes[I]; J++)
            V_Graph[I][J] = '*';

    // Clean the buffer so getchar works.
    getline(cin, Trash);
}

```



```
// Method to display class populations on a horizontal graph
void horizontal_graph(int Dept , vector <int> Classes, vector <vector <char>> V_Graph)
{
    int I,J;
    system("CLS");
    cout << "Department" << setw(40) << "Students" << endl << endl;
    for (I = 0; I < V_Graph.size(); I++)
    {
        cout << Dept << "0" << I << " ";
        for (J = 0; J < V_Graph[0].size(); J++)
            cout << V_Graph[I][J];
        cout << " " << Classes[I] << endl << endl;
    }
    cout << endl << endl;
    cout << "Press ENTER to see vertical graph.";
    getchar();
}

// Method to display class populations on a vertical graph
void vertical_graph(int Dept, vector <int> Classes, vector <vector <char>> V_Graph)
{
    int I,J;
    system("CLS");
    cout<<resetiosflags(ios::adjustfield)<<setiosflags(ios::left)<<setw(15)<<"Students";
    // Display the ten's place number of students
    for (I = 0; I < V_Graph.size(); I++)
        if ((Classes[I] / 10) > 0)
            cout << (Classes[I] / 10) << " ";
        else
            cout << " ";
    cout << endl;
    cout << setw(15) << "";
    // Display the one's place number of students
    for (I = 0; I < V_Graph.size(); I++)
        cout << (Classes[I] % 10) << " ";
    cout << endl;

    // Display the * representing students in each class
    for (I = V_Graph[0].size()-1; I >= 0; I--)
    {
        cout << setw(15) << "";
        for (J = 0; J < V_Graph.size(); J++)
            cout << V_Graph[J][I] << " ";
        cout << endl;
    }
    cout << endl << setiosflags(ios::left) << setw(15) << "Department";
    // Display the Department's number
    for (I = 0; I < V_Graph.size(); I++)
        cout << Dept << " ";
    cout << endl << setw(15) << " ";
    for (I = 0; I < V_Graph.size(); I++)
        cout << "0 ";
    cout << endl << setw(15) << " ";
    for (I = 0; I < V_Graph.size(); I++)
        cout << I << " ";
    cout << endl << endl;
}
```