TERM PROJECT:

WINTER 2023

Course Code: CPS 188

Instructor: Denis Hamelin
Date of Submission: April 2nd, 2023

Names, Student Numbers, and Sections:
Haran Muruganantham
- 501166674, Section 9
Om Balar
- 501165749, Section 9
Shawn Truong
- 501183093, Section 8
Mohammed Sidi
- 501186994, Section 2

This code's goal is to examine and derive conclusions from data gathered by Statistics Canada (Canada's national statistical office) on the prevalence of diabetes in Canada's four most populous provinces which are; Ontario, Quebec, Alberta, and British Columbia. The data was gathered throughout 2015 up until 2021, it contains the percentages of the population with diabetes in each province and the nation as a whole, specifically for ages 35 and above. To assist academics and healthcare professionals in better understanding the diabetes situation in Canada and any potential risk factors related to it. This code will run calculations and offer insights based on the gathered data in an organized matter.

Calculations:

To complete the various calculations required in question 1, conditionals, loops, comparisons, and arrays were used to create the code. To complete the first(a) and the second part(b) of question 1, if-else statements were held inside a for loop that would run through the entire length of data. These if-else statements were set with conditionals that would only allow it to run if it was a specific province. Once meeting a conditional, the corresponding percentage value would be found and summed into an initially empty variable. Also, a counter variable would be used to indicate the amount of data points per province. Once the loop had run through all the data and summed up all the data for each province, each province's sum was divided by its corresponding counter to give an average. For parts three(c) and four(d) of question 1, various arrays were set up due to the number of values being stored and calculated. As a for-loop runs through all the data, the first set of conditionals would check the year or age group, and a corresponding value would be set. This value would allow the data to go to the correct position of the array. Then another set of conditionals was used to compare the provinces. Once the province and year/age group were recognized, the data point would get summed into a provincially specific array at a particular year/age-group based position. Similarly, another array was set up to hold counters to keep track of the amount of data points. Once the entire data set had been run through, another for-loop was used to divide each sum by the corresponding counter to create arrays holding all the averages. For all parts of question 1, each data point would be initially checked if it contained a value of zero, and if it did, the point would not be counted nor summed.

To calculate the highest and lowest percentages of provincial diabetes, multiple conditional if-else statements were used for the values to filter through to determine which had the greatest and least magnitude. At the start of the conditional statements, the first province to test is compared to the other three provinces. If the first province is greater than all three other provinces, the first province is deemed to have the highest magnitude. However, if it is not greater than all three, the first province is no longer considered, and it moves on to check the second province similarly. Again, if the second province fails, it similarly checks the last two provinces. To determine if a specific value is greater than another, the greater than comparison

operator is used, and to determine if all conditions are met, the and logic operator is used. A very similar format calculates the lowest percentage, with the only difference being that the greater than operator is changed to the less than operator.

To find which provinces had an overall higher, lower, or equal percentage of people with diabetes than the national average, comparisons were made for each province's percentage against the national average using if-else statements. If the province's percentage was higher than the national average, then state that it was higher, else if the province's percentage was lower than the national average, then state that it was lower, otherwise the province's percentage must be equal to the national average, then state that it was equal.

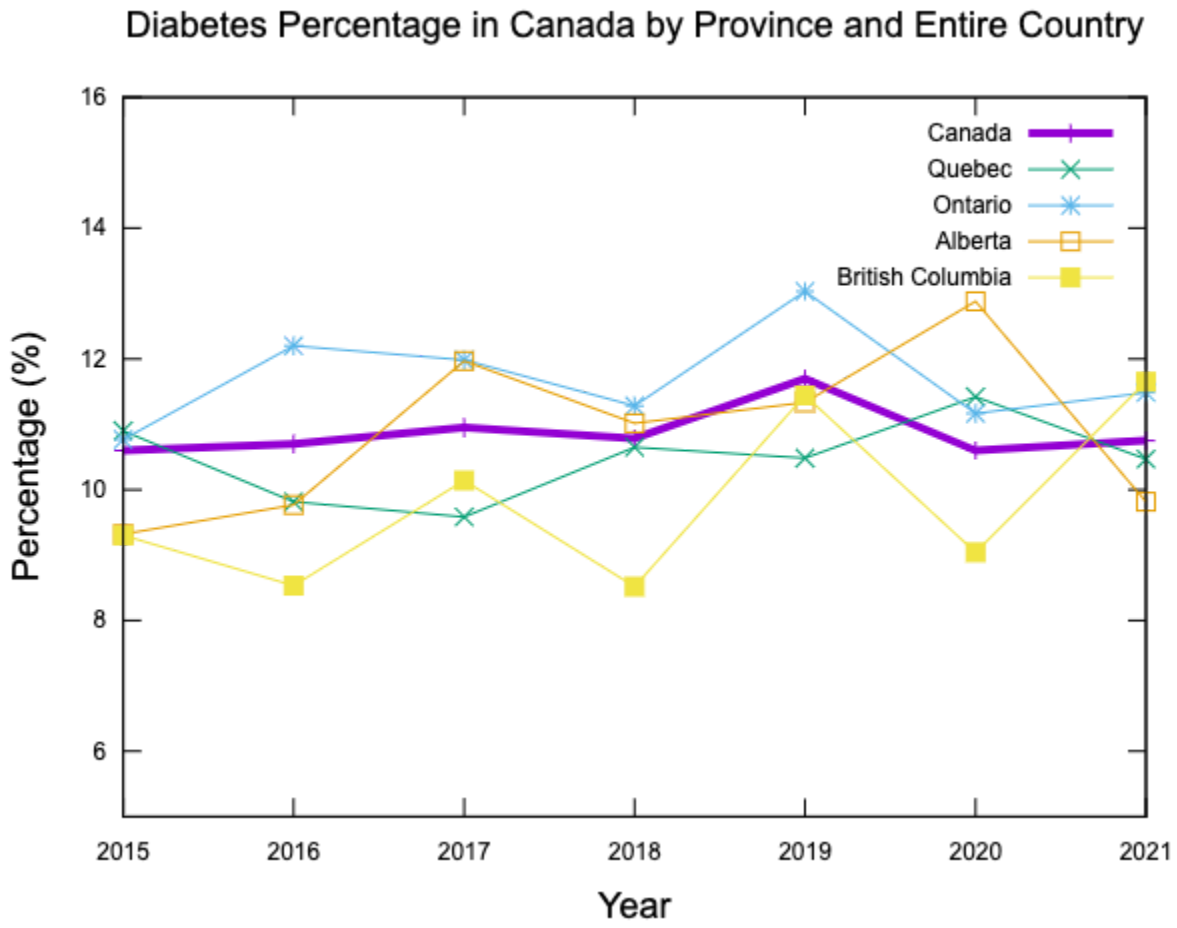
Then, to find which provinces in a certain year had a higher, lower, or equal percentage of people with diabetes than the national average for that year, similar comparisons were made for each province's percentage against the national average using if-else statements. A for-loop was used to go through the arrays containing the yearly average for Ontario, Quebec, Alberta and British Columbia, and compare them with the national average for the year using the same logic stated above.

To determine which province and year had the highest and lowest percentage of diabetes, a struct type pyv was first established to contain all the values associated with a datum (province, year, and percentage value, hence "pyv"). Then, a pyv array of size 28 was made and filled using a for-loop with the required values from the arrays containing a province's yearly averages in 1c), the year, and the province name. Elements 0 to 6 contained the data for Quebec, elements 7 to 13 contained the data for Ontario, elements 14 to 20 contained the data for Alberta, and elements 21 to 27 contained the data for British Columbia. Then, a bubble-sort algorithm was used to sort the pyv array by percentage values. The struct type that was created earlier helps arrange the data by only one relevant value (the percentage), while retaining the other qualities such as year and province when the struct type elements are rearranged. To check for ties between percentage values for the highest and lowest percentage value, for-loops were used. For the lowest percentage tie case, a for-loop was used to iterate through the array and determine how many elements shared the lowest percentage value by recording the index at which the last element had the lowest percentage value through a comparison operator; if the next element has the same percentage value as the first element (known to be the lowest), then record the index and move onto the next element for comparison, otherwise stop checking. A similar for-loop was used to iterate through the array and determine how many elements shared the highest percentage value by recording the index at which the last element had the highest percentage value; starting from element 26, if the element has the same percentage value as the last element (known to be the highest), then record the index and move onto the previous element for comparison, otherwise stop checking. Then, a for-loop was used to print a statement saying a certain year and province had the lowest percentage starting from the 0th element (known to have the lowest

percentage) to the index at which the last element had the lowest percentage. Another for-loop was used to print a statement saying a certain year and province had the highest percentage starting from the 27th element (known to have the highest percentage) to the index at which the last element had the highest percentage.

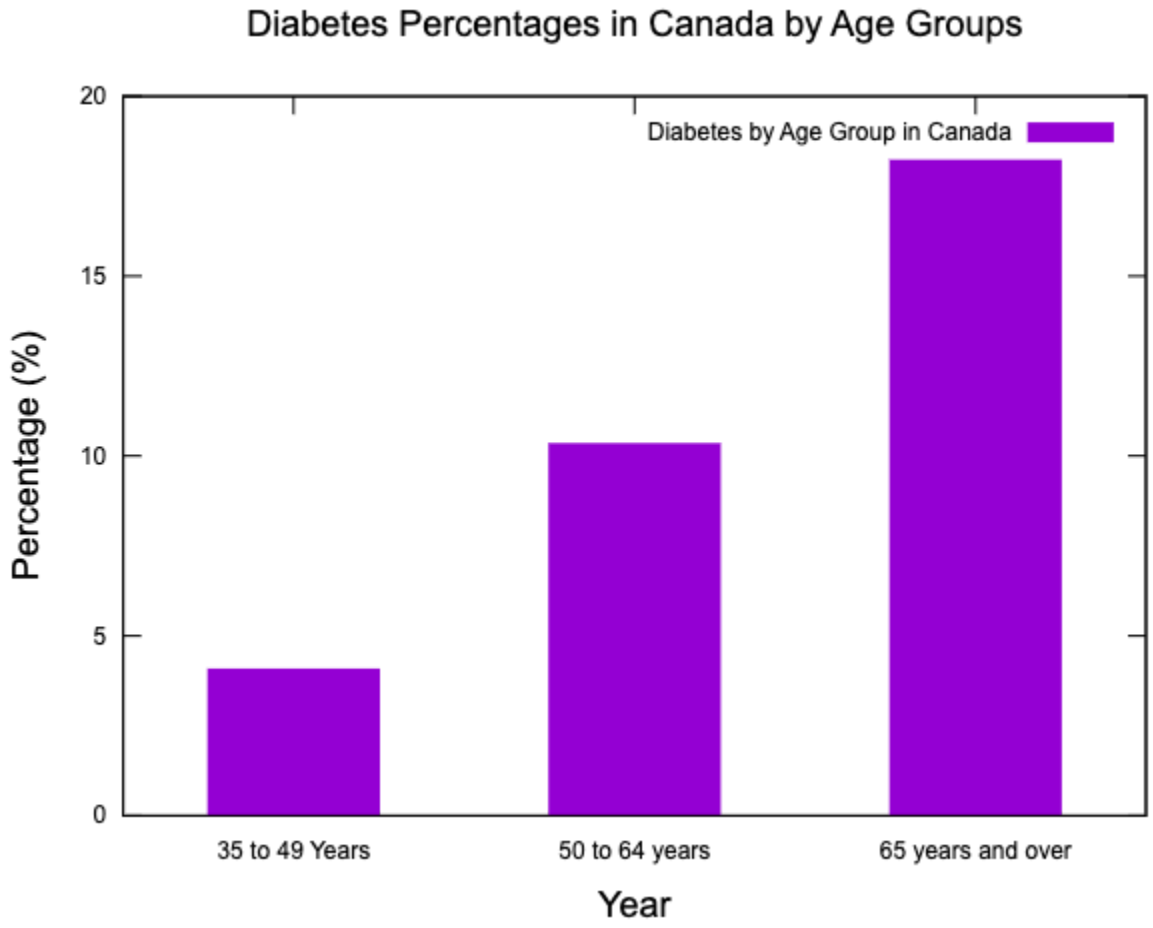
Graphs:

To create the plot, first the data was put into a text file. To do so, a for-loop was used that printed the year and percentage values for each province/country separated by a space. Then the 'set' commands in GNUPlot were used to: set the y-axis scale, the title, y-axis label, x-axis label, and the font-size. Finally, the 'plot' command was used to: get the data from the file, plot it with lines and add a name for each line in the legend. The line width for Canada was set to 4 so that it stands out from the other lines.



Similar to question 5, for this question all the data was also put into a file. To do so, a for-loop was used which went through the country array of diabetes percentages by age group and printed each value to the file. Then in GNUPlot, the "set" command was used to: set the title, set the x-axis label, set the y-axis label, set the bar labels, set the bar widths and style, and set the

y-axis scale. Finally, the “plot” command was used to get the data from the file, plot it, and set a title for it in the legend.



Working on this project has been a great experience. The best way to go into a group based task, is with an open mind. Collaboration is a great way to learn and help each other, while having the same end goal in mind. Our group has proved to be organized and supportive. When ending a group project, feeling more knowledgeable is a sign of a successful project. If we were to repeat this project there is not much we would do differently. We got in contact with one another as soon as possible, and started working as a team to assign parts and set deadlines. Our group managed to create a very efficient work split that pleased everyone. Being able to work together plays a crucial part in having a successful group. If we were to complete this project again, there are a few things we would have done differently. The first is that instead of collecting all the data from the CSV file in multiple arrays, we should have created a struct type that could hold all the relevant data in one row of the CSV. This struct would have contained percentage in a double variable, sex in a string variable (if applicable), age group (if applicable) in a string variable, year in an integer variable, and the location (province or Canada’s average excluding the Territories) in a string variable. During question 4, we noticed that by creating a

struct type, we could have completed the previous questions and question 4 with more ease and clarity. However, given that we had already put in the effort for those questions, we made a struct type specifically to solve question 4 in a simple manner. In addition, we used OnlineGDB as our main source of sharing code in the group. This was inefficient, as OnlineGDB did not let us work on the same file, and it did not provide an easy medium to approve each other's code and facilitate version control in the event that we needed to revert back to a previous version. In hindsight, using GitHub, an online tool that facilitates software version control, would have made the project much simpler to work on as a team. Additionally, for the GNUPlots, instead of creating two files, one for the line graph and one for the bar graph, we would've put all data in one file and used the 'index' command to get the data separately. This way, the program only needs to open 1 file and it keeps everything more organized.

Program Execution:

Question 1a) Average number of diabetic people in each province:

Province	Average
Quebec	10.45 %
Ontario	11.70 %
Alberta	10.86 %
British Columbia	9.67 %

Question 1b) The national average for people with diabetes:
The national average is: 10.87 %

Question 1c) Yearly Averages for each province and whole country:

	Quebec	Ontario	Alberta	British Columbia	Canada (Excluding Territories)
2015	10.90 %	10.77 %	9.32 %	9.30 %	10.60 %
2016	9.82 %	12.20 %	9.77 %	8.53 %	10.70 %
2017	9.58 %	11.98 %	11.97 %	10.14 %	10.95 %
2018	10.65 %	11.28 %	11.02 %	8.52 %	10.78 %
2019	10.48 %	13.03 %	11.33 %	11.44 %	11.70 %
2020	11.42 %	11.17 %	12.88 %	9.04 %	10.60 %
2021	10.47 %	11.48 %	9.82 %	11.65 %	10.75 %

Question 1d) Average number of diabetic people by age group:

	Quebec	Ontario	Alberta	British Columbia	Canada (Excluding Territories)
35 to 49 years	3.35 %	4.64 %	4.46 %	3.43 %	4.06 %
50 to 64 years	9.06 %	11.22 %	10.29 %	7.91 %	10.33 %
65 years and over	18.44 %	19.24 %	16.92 %	15.44 %	18.21 %

Question 2) Province with highest and lowest average:
Ontario has the greatest average
British Columbia has the lowest average

Question 3) Average Comparison with National Average (all years combined):
Quebec had a lower percentage of people with diabetes than the national average.
Ontario had a higher percentage of people with diabetes than the national average.
Alberta had a lower percentage of people with diabetes than the national average.
British Columbia had a lower percentage of people with diabetes than the national average.

Higher and lower than national average by year:

Year	Province	Comparison to National Average
2015	Quebec	Higher than National Average
2015	Ontario	Higher than National Average
2015	Alberta	Lower than National Average
2015	British Columbia	Lower than National Average
2016	Quebec	Lower than National Average
2016	Ontario	Higher than National Average
2016	Alberta	Lower than National Average
2016	British Columbia	Lower than National Average
2017	Quebec	Lower than National Average
2017	Ontario	Higher than National Average
2017	Alberta	Higher than National Average
2017	British Columbia	Lower than National Average
2018	Quebec	Lower than National Average
2018	Ontario	Higher than National Average
2018	Alberta	Higher than National Average
2018	British Columbia	Lower than National Average
2019	Quebec	Lower than National Average
2019	Ontario	Higher than National Average
2019	Alberta	Lower than National Average
2019	British Columbia	Lower than National Average
2020	Quebec	Higher than National Average
2020	Ontario	Higher than National Average
2020	Alberta	Higher than National Average
2020	British Columbia	Lower than National Average
2021	Quebec	Lower than National Average
2021	Ontario	Higher than National Average
2021	Alberta	Lower than National Average
2021	British Columbia	Higher than National Average

Question 4) Year and province with highest and lowest average of diabetic people:
In 2018 British Columbia had the lowest percentage of diabetes.
In 2019 Ontario had the highest percentage of diabetes.

Source Code (GNUPlot):

```
# Question 5

# Type "load "plots.gnu"" in terminal to run file

set title "Diabetes Percentage in Canada by Province and Entire Country"
set title font ",18"
set xlabel "x" font ",18"
set ylabel "y" font ",18"
set xlabel "Year"
set ylabel "Percentage (%)"
set yrange [5:16]
plot       "line_file.txt" using 1:2 title "Canada" with lp lw 4,\
                    "line_file.txt" using 1:3 title "Quebec" with lp,\
                    "line_file.txt" using 1:4 title "Ontario" with lp,\
                    "line_file.txt" using 1:5 title "Alberta" with lp,\
                    "line_file.txt" using 1:6 title "British Columbia" with lp


# Question 6

set title "Diabetes Percentages in Canada by Age Groups"
set xtics ("35 to 49 Years" 1, "50 to 64 years" 2, "65 years and over" 3)
set ylabel "Percentage (%)"
set xlabel "Year"
set boxwidth 0.5
set style fill solid
set yrange [0:20]
plot 'bar_file.txt' using 1:2 title "Diabetes by Age Group in Canada" with boxes ls 1
```

Source Code (C):

```
#include <stdio.h>
#include <string.h>
#include <stdbool.h>
#include <stdlib.h>

#define LARGESTCHAR 1000

// Part of Question 4
typedef struct province_year_value
{
    char province [20];
    int year;
    double percentage;

}pyv;


int main(){

        // Getting data from csv file
        int year[215];
        char province[215][50];
        char age_group[215][20];
        char gender[215][10];
        double percentage[215];

   FILE *fp;
   char buffer[LARGESTCHAR];
   char *line;
   int row = 0;
   int column = 0;

   fp = fopen("statscan_diabetes.csv","r");


   while (feof(fp) != true)
   {
                column = 0;
                row++;

      fgets(buffer, LARGESTCHAR, fp);

      line = strtok(buffer, ",");
```

```
    if(row == 1){
                        continue;
                }

    while(line != NULL)
    {
                        if(column == 0){
                                year[row - 2] = atoi(line);
                        }
                        if (column == 1) {
                                strcpy(province[row - 2], line);

                                if(!strcmp(line, "Canada (excluding territories)")){
                                        column++;
                                }
                        }

                        if (column == 3) {
                                strcpy(age_group[row - 2], line);
                        }

                        if (column == 4) {
                                strcpy(gender[row - 2], line);
                        }
                        if (column == 13) {
                                percentage[row - 2] = atof(line);
                        }

        line = strtok(NULL, ",");
        column++;
    }
}

//Question 1a
double qcSum = 0;
        double onSum = 0;
        double abSum = 0;
        double bcSum = 0;
        int qcCounter = 0;
        int onCounter = 0;
        int abCounter = 0;
        int bcCounter = 0;

        for (int i = 0; i < 210; i++)
```

```
{
        if (strcmp(province[i], "Quebec") == 0)
        {
                if (percentage[i] != 0)
                {
                   double pValue = percentage[i];
                qcSum = qcSum + pValue;
                   qcCounter = qcCounter + 1;
                }
        }
        else if (strcmp(province[i], "Ontario") == 0)
        {
           if (percentage[i] != 0)
                {
                   double pValue = percentage[i];
                onSum = onSum + pValue;
                onCounter = onCounter + 1;
                }
        }
        else if (strcmp(province[i], "Alberta") == 0)
        {
                if (percentage[i] != 0)
                {
                   double pValue = percentage[i];
                   abSum = abSum + pValue;
                   abCounter = abCounter + 1;
                }
        }
        else if (strcmp(province[i], "British Columbia") == 0)
        {
                if (percentage[i] != 0)
                {
                   double pValue = percentage[i];
                bcSum = bcSum + pValue;
                bcCounter = bcCounter + 1;
                }
        }
}

double qcAverage =  qcSum / qcCounter;
double onAverage =  onSum / onCounter;
double abAverage =  abSum / abCounter;
double bcAverage =  bcSum / bcCounter;
```

```
printf("Question 1a) Average number of diabetic people in each province: \n");
printf("Province\t\tAverage\n");
printf("--------\t\t-------\n");
printf("Quebec\t\t\t%.2lf %% \n", qcAverage);
printf("Ontario\t\t\t%.2lf %% \n", onAverage);
printf("Alberta\t\t\t%.2lf %% \n", abAverage);
printf("British Columbia\t%.2lf %% \n", bcAverage);

//Question 1b
double caSum = 0;
int caCounter = 0;

for (int z = 0; z < 210; z++)
{
        if (strcmp(province[z], "Canada (excluding territories)") == 0)
        {
           if (percentage[z] != 0)
                {
                   double pValue = percentage[z];
                caSum = caSum + pValue;
                   caCounter = caCounter + 1;
                }
        }
}

double caAverage = caSum / caCounter;

printf("\nQuestion 1b) The national average for people with diabetes: \n");
printf("The national average is: %.2lf %% \n", caAverage);

//Question 1c
double qcYearlySum[7];
double onYearlySum[7];
double abYearlySum[7];
double bcYearlySum[7];
double caYearlySum[7];

double qcYearlyAvg[7];
double onYearlyAvg[7];
double abYearlyAvg[7];
double bcYearlyAvg[7];
double caYearlyAvg[7];

int qcCount[7];
```

```
    int onCount[7];
    int abCount[7];
    int bcCount[7];
    int caCount[7];

    for (int i = 0; i < 8; i++){
            qcYearlySum[i] = 0;
            onYearlySum[i] = 0;
            abYearlySum[i] = 0;
            bcYearlySum[i] = 0;
            caYearlySum[i] = 0;
            qcCount[i] = 0;
            onCount[i] = 0;
            abCount[i] = 0;
            bcCount[i] = 0;
            caCount[i] = 0;
    }

    int yearNum;
    double pValue;

    for (int j = 0; j < 210; j++)
    {
if (year[j] == 2015)
        {
           yearNum = 0;
        }
        else if (year[j] == 2016)
        {
           yearNum = 1;
        }
        else if (year[j] == 2017)
        {
           yearNum = 2;
        }
        else if (year[j] == 2018)
        {
           yearNum = 3;
        }
        else if (year[j] == 2019)
        {
           yearNum = 4;
        }
        else if (year[j] == 2020)
```

```
    {
        yearNum = 5;
    }
    else if (year[j] == 2021)
    {
        yearNum = 6;
    }


    if (strcmp(province[j], "Quebec") == 0)
{
            if (percentage[j] != 0)
            {
                pValue = percentage[j];
                qcYearlySum[yearNum] += + pValue;
                qcCount[yearNum] = qcCount[yearNum] + 1;
            }
}
else if (strcmp(province[j], "Ontario") == 0)
{
    if (percentage[j] != 0)
    {
                pValue = percentage[j];
                onYearlySum[yearNum] = onYearlySum[yearNum] + pValue;
                onCount[yearNum] = onCount[yearNum] + 1;
    }
}
else if (strcmp(province[j], "Alberta") == 0)
{
    if (percentage[j] != 0)
    {
                pValue = percentage[j];
                abYearlySum[yearNum] = abYearlySum[yearNum] + pValue;
                abCount[yearNum] = abCount[yearNum] + 1;
    }
}
else if (strcmp(province[j], "British Columbia") == 0)
{
    if (percentage[j] != 0)
    {
            pValue = percentage[j];
            bcYearlySum[yearNum] = bcYearlySum[yearNum] + pValue;
            bcCount[yearNum] = bcCount[yearNum] + 1;
    }
```

```
        }
        else if (strcmp(province[j], "Canada (excluding territories)") == 0)
        {
           if (percentage[j] != 0)
           {
              pValue = percentage[j];
                    caYearlySum[yearNum] = caYearlySum[yearNum] + pValue;
                    caCount[yearNum] = caCount[yearNum] + 1;
           }
        }
        //printf("%lf\n", pValue);
        }

        for (int k = 0; k < 7; k++)
        {
           qcYearlyAvg[k] = qcYearlySum[k] / qcCount[k];
           onYearlyAvg[k] = onYearlySum[k] / onCount[k];
           abYearlyAvg[k] = abYearlySum[k] / abCount[k];
           bcYearlyAvg[k] = bcYearlySum[k] / bcCount[k];
           caYearlyAvg[k] = caYearlySum[k] / caCount[k];
        }

        int firstYear = 2015;

        printf("\nQuestion 1c) Yearly Averages for each province and whole country: \n");
        printf("\tQuebec\t\tOntario\t\tAlberta\t\tBritish Columbia\t\tCanada (Excluding Territories)\n");
        printf("\t------\t\t-------\t\t-------\t\t----------------\t\t------------------------------\n");
        for (int l = 0; l < 7; l++)
        {
          printf("%d\t%.2lf %%\t\t%.2lf %%\t\t%.2lf %%\t\t%.2lf %%\t\t\t%.2lf %% \n", (firstYear + l),
qcYearlyAvg[l], onYearlyAvg[l], abYearlyAvg[l], bcYearlyAvg[l], caYearlyAvg[l]);

        }

        //Question 1d
        double qcAgeSum[3];
        double onAgeSum[3];
        double abAgeSum[3];
        double bcAgeSum[3];
        double caAgeSum[3];

        double qcAgeAvg[3];
        double onAgeAvg[3];
        double abAgeAvg[3];
```

```
    double bcAgeAvg[3];
    double caAgeAvg[3];

    int qcElements[3];
    int onElements[3];
    int abElements[3];
    int bcElements[3];
    int caElements[3];

    for (int i = 0; i < 4; i++){
            qcAgeSum[i] = 0;
            onAgeSum[i] = 0;
            abAgeSum[i] = 0;
            bcAgeSum[i] = 0;
            caAgeSum[i] = 0;
            qcElements[i] = 0;
            onElements[i] = 0;
            abElements[i] = 0;
            bcElements[i] = 0;
            caElements[i] = 0;

    }

    int ageNum;

    for (int r = 0; r < 210; r++)
    {
if (strcmp(age_group[r], "35 to 49 years") == 0)
      {
        ageNum = 0;
      }
      else if (strcmp(age_group[r], "50 to 64 years") == 0)
      {
        ageNum = 1;
      }
      else if (strcmp(age_group[r], "65 years and over") == 0)
      {
        ageNum = 2;
      }


      if (strcmp(province[r], "Quebec") == 0)
    {
      if (percentage[r] != 0)
```

```
    {
        double pValue = percentage[r];
        qcAgeSum[ageNum] = qcAgeSum[ageNum] + pValue;
        qcElements[ageNum] = qcElements[ageNum] + 1;
    }
}
else if (strcmp(province[r], "Ontario") == 0)
{
if (percentage[r] != 0)
    {
        double pValue = percentage[r];
        onAgeSum[ageNum] = onAgeSum[ageNum] + pValue;
          onElements[ageNum] = onElements[ageNum] + 1;
    }
}
else if (strcmp(province[r], "Alberta") == 0)
{
        if (percentage[r] != 0)
    {
        double pValue = percentage[r];
        abAgeSum[ageNum] = abAgeSum[ageNum] + pValue;
          abElements[ageNum] = abElements[ageNum] + 1;
    }
}
else if (strcmp(province[r], "British Columbia") == 0)
{
        if (percentage[r] != 0)
    {
        double pValue = percentage[r];
        bcAgeSum[ageNum] = bcAgeSum[ageNum] + pValue;
          bcElements[ageNum] = bcElements[ageNum] + 1;
    }
}
else if (strcmp(province[r], "Canada (excluding territories)") == 0)
{
    if (percentage[r] != 0)
    {
        double pValue = percentage[r];
        caAgeSum[ageNum] = caAgeSum[ageNum] + pValue;
          caElements[ageNum] = caElements[ageNum] + 1;
    }
}
}
```

```
        for (int d = 0; d < 3; d++)
        {
           qcAgeAvg[d] = qcAgeSum[d] / qcElements[d];
           onAgeAvg[d] = onAgeSum[d] / onElements[d];
           abAgeAvg[d] = abAgeSum[d] / abElements[d];
           bcAgeAvg[d] = bcAgeSum[d] / bcElements[d];
           caAgeAvg[d] = caAgeSum[d] / caElements[d];
        }

        printf("\nQuestion 1d) Average number of diabetic people by age group: \n");
        printf("\t\t\tQuebec\t\tOntario\t\tAlberta\t\tBritish Columbia\tCanada (Excluding Territories)\n");
        printf("\t\t\t------\t\t-------\t\t-------\t\t----------------\t------------------------------\n");

        for (int s = 0; s < 3; s++)
        {
          printf("%s  \t%.2lf %%\t\t%.2lf %%\t\t%.2lf %%\t\t%.2lf %%   \t\t%.2lf %% \n",
age_group[14*s], qcAgeAvg[s], onAgeAvg[s], abAgeAvg[s], bcAgeAvg[s], caAgeAvg[s]);
        }

        //Question 2
        printf("\nQuestion 2) Province with highest and lowest average: \n");
        if (qcAverage > onAverage && qcAverage > abAverage && qcAverage > bcAverage)
        {
                printf("Quebec has the greatest average \n");
        }
        else if (onAverage > abAverage && onAverage > bcAverage)
        {
           printf("Ontario has the greatest average \n");
        }
        else if (abAverage > bcAverage)
        {
                printf("Alberta has the greatest average \n");
        }
        else
        {
                printf("British Columbia has the greatest average \n");
        }

        if (qcAverage < onAverage && qcAverage < abAverage && qcAverage < bcAverage)
        {
                printf("Quebec has the lowest average");
        }
        else if (onAverage < abAverage && onAverage < bcAverage)
        {
```

```
        printf("Ontario has the lowest average");
}
else if (abAverage < bcAverage)
{
        printf("Alberta has the lowest average");
}
else
{
        printf("British Columbia has the lowest average");
}

// Question 3
printf("\n\nQuestion 3) Average Comparison with National Average (all years combined): \n");

//check quebec
if (qcAverage > caAverage){
   printf("Quebec had a higher percentage of people with diabetes than the national average.\n");
}
else if(qcAverage < caAverage){
   printf("Quebec had a lower percentage of people with diabetes than the national average.\n");
}
else{
   printf("Quebec had the same percentage of people with diabetes as the national average.\n");
}
//check ontario
if (onAverage > caAverage){
   printf("Ontario had a higher percentage of people with diabetes than the national average.\n");
}
else if(onAverage < caAverage){
   printf("Ontario had a lower percentage of people with diabetes than the national average.\n");
}
else{
   printf("Ontario had the same percentage of people with diabetes as the national average.\n");
}
//check alberta
if (abAverage > caAverage){
   printf("Alberta had a higher percentage of people with diabetes than the national average.\n");
}
else if(abAverage < caAverage){
   printf("Alberta had a lower percentage of people with diabetes than the national average.\n");
}
else{
   printf("Alberta had the same percentage of people with diabetes as the national average.\n");
}
```

```
        //check british columbia
        if (bcAverage > caAverage){
            printf("British Columbia had a higher percentage of people with diabetes than the national
average.\n\n");
        }
        else if(bcAverage < caAverage){
            printf("British Columbia had a lower percentage of people with diabetes than the national
average.\n\n");
        }
        else{
            printf("British Columbia had the same percentage of people with diabetes as the national
average.\n\n");
        }


    printf("Higher and lower than national average by year: \n");

    firstYear = 2015;

    printf("Year\t\tProvince\t\tComparison to National Average\n");
    printf("----\t\t--------\t\t------------------------------\n");

    // check for each year
          for (int l = 0; l < 7; l++)
          {
            //quebec
      if (qcYearlyAvg[l] > caYearlyAvg[l]){
         printf("%d\t\tQuebec\t\t\tHigher than National Average\n", (firstYear + l));
      }
      else if(qcYearlyAvg[l] < caYearlyAvg[l]){
         printf("%d\t\tQuebec\t\t\tLower than National Average\n", (firstYear + l));
      }
      else{
         printf("%d\t\tQuebec\t\t\tSame as National Average\n", (firstYear + l));
      }

      //ontario
      if (onYearlyAvg[l] > caYearlyAvg[l]){
         printf("%d\t\tOntario\t\t\tHigher than National Average\n", (firstYear + l));
      }
      else if(onYearlyAvg[l] < caYearlyAvg[l]){
         printf("%d\t\tOntario\t\t\tLower than National Average\n", (firstYear + l));
      }
```

```
    else{
        printf("%d\t\tOntario\t\t\tSame as National Average\n", (firstYear + l));
    }

    //alberta
    if (abYearlyAvg[l] > caYearlyAvg[l]){
        printf("%d\t\tAlberta\t\t\tHigher than National Average\n", (firstYear + l));
    }
    else if(abYearlyAvg[l] < caYearlyAvg[l]){
        printf("%d\t\tAlberta\t\t\tLower than National Average\n", (firstYear + l));
    }
    else{
        printf("%d\t\tAlberta\t\t\tSame as National Average\n", (firstYear + l));
    }

    //british columbia
    if (bcYearlyAvg[l] > caYearlyAvg[l]){
        printf("%d\t\tBritish Columbia\tHigher than National Average\n\n", (firstYear + l));
    }
    else if(bcYearlyAvg[l] < caYearlyAvg[l]){
        printf("%d\t\tBritish Columbia\tLower than National Average\n\n", (firstYear + l));
    }
    else{
        printf("%d\t\tBritish Columbia\tSame as National Average\n\n", (firstYear + l));
    }
        }

        // Question 4
        printf("Question 4) Year and province with highest and lowest average of diabetic people: \n");

pyv all_values[28];

for(int i = 0; i < 7; i++){
    all_values[i].percentage = qcYearlyAvg[i];
    all_values[i].year = firstYear + i;
    strncpy(all_values[i].province, "Quebec", 7);

    all_values[i + 7].percentage = onYearlyAvg[i];
    all_values[i + 7].year = firstYear + i;
    strncpy(all_values[i + 7].province, "Ontario", 8);

    all_values[i + 14].percentage = abYearlyAvg[i];
    all_values[i + 14].year = firstYear + i;
    strncpy(all_values[i + 14].province, "Alberta", 8);
```

```
    all_values[i + 21].percentage = bcYearlyAvg[i];
    all_values[i + 21].year = firstYear + i;
    strncpy(all_values[i + 21].province, "British Columbia", 17);
  }

  for(int i = 0; i < 28 - 1; i++){
    for(int j = 0; j < 28 - i - 1; j++){
      if(all_values[j].percentage >= all_values[j+1].percentage){
        pyv temp = all_values[j+1];
        all_values[j+1] = all_values[j];
        all_values[j] = temp;
      }
    }
  }

  int max_lowest = 0;
  int max_highest = 27;

  for(int i = 0; i < 28 - 1; i++){
    if(all_values[i + 1].percentage == all_values[0].percentage){
      max_lowest += 1;
    }
    else{
      break;
    }
  }

       for(int i = 27; i > 1; i--){
         if(all_values[i - 1].percentage == all_values[27].percentage){
           max_highest -= 1;
         }
         else{
           break;
         }
       }

  for(int i = 0; i <= max_lowest; i++){
         printf("In %d %s had the lowest percentage of diabetes.\n", all_values[i].year,
all_values[i].province);
       }
       for(int i = 27; i >= max_highest; i--){
         printf("In %d %s had the highest percentage of diabetes.\n", all_values[i].year,
all_values[i].province);
```

```
        }

        // Adding data to file for Question 5
        fp = fopen("line_file.txt", "w");

    for (int i = 2015; i <= 2021; i++){
                fprintf(fp, "%d %lf %lf %lf %lf %lf\n", i, caYearlyAvg[i - 2015], qcYearlyAvg[i - 2015],
onYearlyAvg[i - 2015], abYearlyAvg[i - 2015], bcYearlyAvg[i - 2015]);
        }
        fprintf(fp, "\n");
        fclose(fp);

        // Adding data to file for Question 6
    fp = fopen("bar_file.txt", "w");

    for (int i = 1; i < 4; i++){
                fprintf(fp, "%d %lf\n", i, caAgeAvg[i - 1]);
        }
        fprintf(fp, "\n");
        fclose(fp);

    return 0;
}
```