

CPS LAB #4

Section 05

501175325 Krunal Patel

Problem 1

```
#include <stdio.h>

int main()
{
    /**Stating the variables for the loops, the amount of rows present and the coef values in the
rows**/
    int rows = 9;
    int coef = 1;

    /**First loop created that loops for the amount of rows present**/
    for(int i = 0; i < rows; i++)
    {
        /**Second loop created**/
        for(int space = 1; space <= rows - i; space++)
        {

            printf(" ");
        }
        /**Third loop prints the values present in each of the rows**/
        for(int j = 0; j <= i; j++)
        {
            /**This path followed by printing 1 if the value in the first row is being found**/
            if (j == 0 || i == 0)
            {

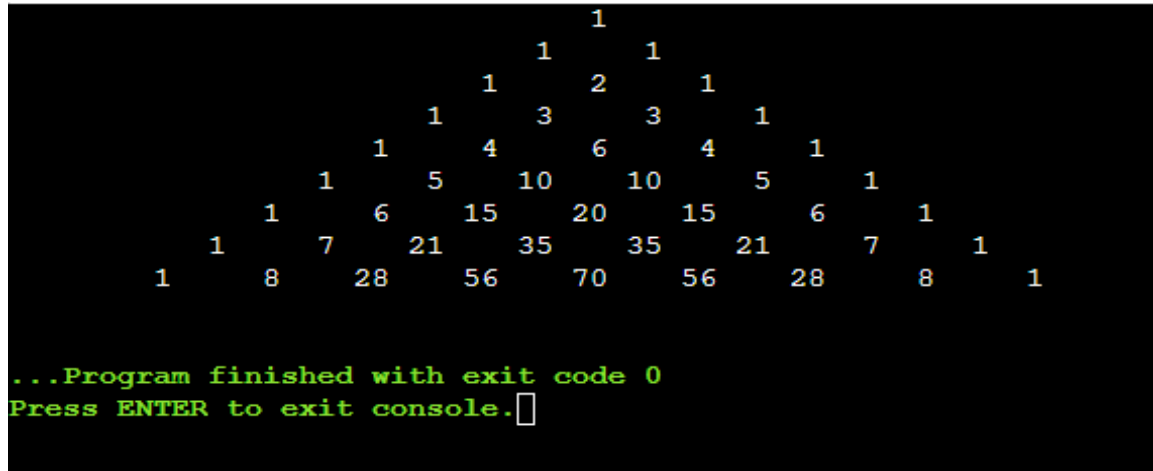
                coef = 1;
                /**This path followed if the value is not being found in the first row**/
            } else
            {
                /**Program solves for the coef values in each one of the rows present**/
                coef = coef * (i - j + 1) / j;
            }
        }
        /**The values from the rows are printed**/
        printf("%6d", coef);
```

```

    }
    /*The next row is printed on a new line by using "\n"*/
    printf("\n");
}

return 0;
}

```



```

          1
        1 1
      1 2 1
    1 3 3 1
  1 4 6 4 1
1 5 10 10 5 1
  1 6 15 20 15 6 1
    1 7 21 35 35 21 7 1
      1 8 28 56 70 56 28 8 1

...Program finished with exit code 0
Press ENTER to exit console.

```

Problem 2

Algorithm:

1. Start the code
2. Next, state the variables that will be used within the program for elements of the program such as loops, employee number, shifts, base pay, per-shift hours worked, and total hours worked, as well as the total pay
3. Now, create an array that is able to store the different elements of the program such as the number of shifts worked, employee number, hourly rate, and hours worked per shift.
4. Moving on, calculate the number of employees that are within this array. This can be done by taking the total size of the array and then dividing it by the first value that is in each row.
5. Next, find the total hours worked by finding the sum of the hours worked for each shift.
6. Now, calculate the total pay the workers receive by multiplying the total hours they work by the base hourly rate and the premium rate for the number of hours worked as well.
7. Lastly, make a formatted table and print out the employee number, total hours worked, and rounded total pay. The formatted table will consist of these three elements.
8. Now, simply complete the array by repeating steps 4-7 for each of the workers.
9. End the code

Code:

```
#include <stdio.h>

int main()
{
    /**Declare variables used in the program*/
    int j, i, emp_num, numshift;
    double rate, hours, totalh = 0, totalp = 0;

    /**Create a 2D array to store employee data*/
    int employee_data[][20] = {
        {77621, 6, 18.00, 6, 7, 5, 2, 7, 8},
        {82010, 3, 22.50, 7, 3, 6},
        {92390, 8, 19.50, 4, 8, 4, 7, 2, 7, 8, 6},
        {62396, 2, 32.00, 6, 6},
        {89320, 1, 27.50, 9},
        {19089, 7, 16.00, 5, 6, 7, 8, 9, 2, 6},
        {54209, 11, 17.00, 8, 9, 2, 5, 8, 9, 2, 5, 6, 8, 2},
        {50630, 4, 20.00, 8, 8, 8, 8}
    };

    /**Calculate the number of employees in the array*/
    int employees = sizeof(employee_data) / sizeof(employee_data[0]);

    /**Print column headers*/
    printf("Employee Number    Total Hours    Total Pay\n");
    printf("-----\n");

    /**Loop through the employee data array to calculate total hours and total pay for each employee*/
    for (i = 0; i < employees; i++) {
        /**Extract employee number, number of shifts worked and hourly rate from the array*/
        emp_num = employee_data[i][0];
        numshift = employee_data[i][1];
        rate = employee_data[i][2];

        /**Loop through the shifts worked by each employee to calculate total hours worked*/
        for (j = 0; j < numshift; j++)
        {
```

```

        hours = employee_data[i][j+3];
        /*add hours to total hours worked*/
        totalh += hours;
    }

    /*Calculate total pay based on total hours worked*/
    if (totalh >= 15 && totalh <= 25)
    {
        // apply a 5% premium for hours worked between 15 and 25*/
        totalp = totalh * (rate * 1.05);
    } else if (totalh > 25)
    { /*apply a 10% premium for hours worked over 25*/
        totalp = totalh * (rate * 1.1);
    } else
    { /*no premium for hours worked under 15*/
        totalp = totalh * rate;
    }

    /*Print employee number, total hours worked and total pay in columns*/
    printf("%d          %0.1f          $%0.2fn", emp_num, totalh, totalp);

    /*Reset total hours and total pay for the next employee*/
    totalh = 0;
    totalp = 0;
}

return 0;
}

```

Employee Number	Total Hours	Total Pay
77621	35.0	\$693.00
82010	16.0	\$369.60
92390	46.0	\$961.40
62396	12.0	\$384.00
89320	9.0	\$243.00
19089	43.0	\$756.80
54209	64.0	\$1196.80
50630	32.0	\$704.00

Problem 3

```
#include <stdio.h>

int main()
{
    /*Declaring and then initialize variables*/
    double initial_pressure = 50;
    double initial_temperature = 300;
    double final_pressure = 500;
    double final_temperature;
    double pressure;
    double temperature = 273.15;

    /*Calculating the final temperature using the initial conditions that are known*/
    final_temperature = (final_pressure * initial_temperature) / initial_pressure;

    /*Print the headers of the table headers which will be displayed in the output*/
    printf("Temperature (K)\tPressure (atm)\n");
    printf("-----\t-----\n");

    /*Calculating the pressure at the initial temperature and then printing the result*/
    pressure = (initial_pressure * temperature) / initial_temperature;
    printf("%0.2f\t%0.2f\n", temperature, pressure);

    /*A loop is created which will only break when the pressure exceeds the maximum pressure*/
    while (pressure <= final_pressure) {
        /*Increase temperature by 100*/
        temperature += 100;

        /*Now calculate pressure at the new temperature*/
        pressure = (initial_pressure * temperature) / initial_temperature;

        /*Checking if the temperature has exceeded the final temperature or not*/
        if (temperature >= final_temperature) {
            /*The program prints "Kaboom!" and then exits the loop afterwards*/
            printf("Kaboom!\n");
            break;
        }
    }
}
```

```
    /*Now the temperature and pressure is printed*/  
    printf("%.2f\t\t%.2f\n", temperature, pressure);  
}  
  
return 0;  
}
```

```
Temperature (K) Pressure (atm)  
-----  
273.15      45.52  
373.15      62.19  
473.15      78.86  
573.15      95.53  
673.15     112.19  
773.15     128.86  
873.15     145.53  
973.15     162.19  
1073.15    178.86  
1173.15    195.53  
1273.15    212.19  
1373.15    228.86  
1473.15    245.53  
1573.15    262.19  
1673.15    278.86  
1773.15    295.52  
1873.15    312.19  
1973.15    328.86  
2073.15    345.52  
2173.15    362.19  
2273.15    378.86  
2373.15    395.52  
2473.15    412.19  
2573.15    428.86  
2673.15    445.52  
2773.15    462.19  
2873.15    478.86  
2973.15    495.52  
Kaboom!  
  
...Program finished with exit code 0  
Press ENTER to exit console.□
```