

TRIBHUVAN UNIVERSITY INSTITUTE OF ENGINEERING

Pulchowk Campus



A Course Project Submitted to the Department of Electronics and Computer Engineering in partial fulfillment of the requirements for the practical course on Computer Programming [CT 401].

DIGITAL CLOCK

SUBMITTED TO:

The Department of Electronics and
Computer Engineering,
Pulchowk Campus, Lalitpur
Institute of Engineering

SUBMITTED BY:

Pyush Kunwar (077BCE114)
Rana Sah (077BCE122)
Samyog Bhattarai (077BCE139)
Sansar Sah (077BCE142)

Table of Contents

Acknowledgement.....	3
Introduction.....	4
Background	4
Objectives.....	5
Limitations.....	5
General theory.....	6
Problem Analysis.....	11
Algorithm.....	12
Flowchart.....	15
Source code.....	26
Output.....	39
Conclusion.....	40

ACKNOWLEDGEMENT

We the students of BCE E/F have created a simple mini-project called “**Digital Clock**” using C-programming. We would like to thank our subject teacher **Mr. Anup Shrestha** for providing us with this opportunity to create a project on your own using all the knowledge and information that we have gained throughout this semester. This project insisted us into research works as we needed help form various sources like google, different books etc. without which this project wouldn't have been any successful. We would like to offer our deep thanks to all those who were involved in helping us with this project

We tried our best to make this project user-friendly and simple as possible. This is meant for everyone who has general knowledge of English language to perform various activities. This is just a prototype and hence many bugs and errors may arise; so, pardon us for such errors. We will surely fix them in future updates.

The main objective of this program is to illustrate how a digital clock works with the help of C Programming. This program enables the administrator or user to input the computer time and set a digital clock. Different concepts of functions, loops and switch statements are used in this project.

We also consulted various book for theoretical parts of this project. The book below mentioned helped us a lot for the theory, flowchart, algorithm and source code part. Some of them are listed below:

- Mr. Venugopal-Programming With C.
- Mr.Yashwant Kanetkar-Let Us C.
- Mr. Byron S. Gotterfried-Programming With C.
- Mr.Balagurusamy-Programming In ANSI C

Introduction

This mini project in C is aimed towards efficient, easy and affordable illustration of a Digital Clock. This program can eliminate all the drawbacks in the tedious process of creating an analog clock. Apart from that computerized system is the necessity of modern day organizations. So, this project has useful practical applications.

Switch statements, loops and gotoxy functions form the core of this project. It is a simple application program written in C programming language, and compiled using GNU GCC compiler on Code:Blocks IDE. The source code consists of over 275 lines of code which are made user friendly as possible by the optimum use of functions to make the function discrete and divide the project coding among the group members. The project source code effectively utilizes the user defined functions and the concept of switch statements.

Background and problem statement

Engineering being a technical and applied field of study, the most important factor for students is to measure their skills, ability and understandings in practical approach rather than theoretical knowledge. Considering this fact a project is given to students of Civil Engineering-I(1st year 1st part) to prepare a program in C language to reinforce our insights and understandings gained during the lab and theory classes of computer programming[CT401] as per requirement.

This project is a part of the requirement of practical course on computer programming for Civil 1st year 1st part students.

This project that we have worked on is based on switch statements and loops. We decided to do project on this topic as it not only utilizes the core concepts taught in the curriculum. Also this program has practical applications.

To build this project it took us nearly a week managing time between our day to day classes. The various theory and lab classes conducted by our lecturer **SG sir** throughout

the semester proved to be very helpful. Apart from that we consulted various books and internet in gaining concepts of computer programming.

After gaining the concepts and imagining a layout and design of our program, we divided the workload among us and started writing the codes. After successfully writing, compiling, linking and executing the program we then debugged some errors that were occurred during programming. This helped a lot to get insight at debugging too. Then we divided the workload for writing report and at last combined it too. It was also very helpful in developing our skill in team working.

Objectives

1. To explore the features of C Programming
2. To be familiar with resource reusability by making user defined functions.
3. To make the program to occupy less memory as far as possible.
4. To divide the program into smaller manageable parts using functions and make it more eligible and understandable.
5. To gain insight on debugging.
6. To get basic idea of software development process.
7. To learn to work as a team and share responsibilities.

Limitations

1. The user must enter the computer time in correct format.

General theory

C is structured programming based computer programming language was developed by Dennis Ritchie at Bell laboratories in 1972. C is a general-purpose, imperative computer programming language, supporting structured programming, lexical variable scope and recursion, while a static type system prevents many unintended operations. By design, C provides constructs that map efficiently to typical machine instructions, and therefore it has found lasting use in applications that had formerly been coded in assembly language, including operating systems, as well as various application software for computers ranging from supercomputers to embedded systems.

Control Statement:

Logical operation is carried out by several symmetrical or logical statements. There are two types of control statement based on their function.

Selective structure:

Selective structures are used when we have a number of situations where we need to change the order of execution of statements based on certain condition. The selective statements make a decision to take the right path before changing the order of execution.

C provides the following statements for selective structure:

- i.If statements
- ii.switch statements

if statements:

The if statement is a powerful decision making statement and it is used to control the flow of execution of statements. It is a two way statement and is used in conjunction with an expression.

If statement allows the computer to evaluate the expression first and then on depending whether the value of the expression is true or false it transfer the control to the particular statement. At this point of the program has two paths to follow: one for true condition and other for false condition. The types of if statements are explained below:

Simple if statement:

The simple if statement is used to conditionally excite a block of code based on whether a

test condition is true or false. If the condition is true the block of code is executed, otherwise it is skipped. The syntax of if statement is given below:

```
if(test expression)
{
statement-block;
}
statement-x;
```

if else statement:

Another use of else is when there are multiple conditional statements that may all evaluate to true, yet you want only one if statement's body to execute. You can use an "else if" statement following an if statement and its body; that way, if the first statement is true, the "else if" will be ignored, but if the if statement is false, it will then check the condition for the else if statement. If the if statement was true the else statement will not be checked. It is possible to use numerous else if statements to ensure that only one block of code is executed. The syntax of if else statement is given below:

```
if(test expression)
{ true block statement;
} else {
false block statement;
}
```

The switch statement:

C has built in multi way decision statement known as switch. It successively test the value of an expression against a list of case values (integer or character consonants).when a match is found the statement associated with that case is executed. The syntax of switch expression is given below:

```
switch(expression)
{
case constant-1:
    block-1;
    break;
case constant-2:
```

```

        block-2;
        break;
.....
case constant-n:
        block-n;
        break;
default:
        default statement:
}

```

Looping:

You may encounter situations, when a block of code needs to be executed several number of times. In general, statements are executed sequentially. The first statement in a function is executed first, followed by the second, and so on. Programming languages provide various control structures that allow for more complicated execution paths. A loop statement allows us to execute a statement or group of statements multiple times. Different types of loops are discussed below with their major characteristics and syntax used in C:

while loop:

The “while loop” specifies that a section of code should be executed while a certain condition holds true. The syntax of while loop is given below:

```

while(test expression)
{
    body of loop
    ( statements block) ;
}

```

do while statement:

The do while statement is very similar to while statement. It also specifies that a section of code should be executed while a certain condition holds true. the difference between while and do while loop is that while loop test its condition at the top of its loop but do while loop tests its condition at the bottom of loop. In while loop, if the test condition is false, the block of code is skipped. Since condition is tested at the bottom of loop in do

while loop, its block of code is always executed at least once. The syntax of do while loop is given below:

```
do {  
body of loop  
}while (test expression);
```

For loop:

The for loop is used to execute a block of code for a fixed number of repetitions.

Initialization is generally an assignment statement used to set loop control variable. Test expression is a relational expression that determines when loop exits. Update expression defines how the loop variable changes each time when the loop is repeated. The syntax of for loop is given below:

```
for (initialization expression ; test expression ; update expression)  
{  
body of loop;  
}
```

break statement:

The break statement in C programming has the following two usages –

- When a break statement is encountered inside a loop, the loop is immediately terminated and the program control resumes at the next statement following the loop.
- It can be used to terminate a case in the switch statement (covered in the next chapter).

In case of nested loops, the break statement will stop the execution of the innermost loop and start executing the next line of code after the block.

Function:

A function is a self-contained program segment that carries out some specific well defined task. Every c program consists of one or functions. Execution of program always begins by carrying out instruction in main. Function makes program significantly easier to understand and maintain. A well written function may be reused in multiple programs. Program that are easier to design, debug and maintain.

Return statement:

A function may or may not send back any value to the calling function. If it does, it is through return statement. The called function can only return only one value per call at most. The syntax of return statement is given below:

Return variable;

Or return ;

gotoxy():

gotoxy() is an inbuilt function used in C and C++ language as co-ordinates for printing your text or character on the screen on your desired place. Two parameters are inserted in the bracket-(), that are the value for x-co-ordinate and the second one value for the y-co-ordinate.

sleep():

C programming language provides `sleep()` function in order to wait for a current thread for a specified time.

Problem analysis

Understanding the problem

A meeting of the group members was held and discussion regarding the project was made. First we decided to make project in student record system then discussions and decisions for the: *input, layout, options, logic and output* of the project were made.

Input requirements

The program was made command prompt type. The code must be clear so as to direct the user to input desired data in desired format. The program starts by the user requiring inputting the computer time.

Output requirements

Before programming it should be decided how the output should be displayed. The command prompt window starts clocking the time input by the user.

Processing requirement

The processing requirement should be clearly defined to convert the given data to the required output. In this project four people collaborated together for the completion of project. The coding is done in 64-bit OS using GNU GCC compiler on Code::Blocks IDE.

Technical feasibility

In the context of feasibility the coding didn't require that much hard work and manpower due to simplicity of program. Though the program was short, it was user interactive and it yielded the required output.

Algorithm and flowchart

2.1 Algorithm

A. void call (digit ,x,y)

1. Start
2. If case=1, one(x,y) else Exit
3. If case=2, two(x,y) else Exit
4. If case=3, three (x,y) else Exit
5. If case=4, four (x,y) else Exit
6. If case=5, five(x,y) else Exit
7. If case=6, six(x,y) else Exit
8. If case=7, seven(x,y)else Exit
9. If case=8, seven(x,y) else exit
10. If case=9, eight(x,y) else exit
11. If case=10, nine(x,y) else exit
12. Else zero(x,y)

B.void colon(x,y)

1. Start
2. i = 1
3. if i<=9 increase i by 1
4. then gotoxy(x,y)
5. if i=3 and j=3 OR i=7 and j=3 then print *
6. else print “ “
- 7.increase x by 1 and y by 1
- 8.x=x-5

C.void zero(x,y)

1. Start
2. $i = 1$
3. if $i \leq 9$ increase i by 1
4. then gotoxy(x,y)
5. if $i=1$ OR $i=9$ OR $j=1$ OR $j=5$ then print *
6. else print “ “
- 7.increase x by 1 and y by 1
8. $x=x-5$

D.void one(x,y)

1. Start
2. $i = 1$
3. if $i \leq 9$ increase i by 1
4. then gotoxy(x,y)
5. if $j=1$ then print *
6. else print “ “
- 7.increase x by 1 and y by 1
8. $x=x-5$

E.void two(x,y)

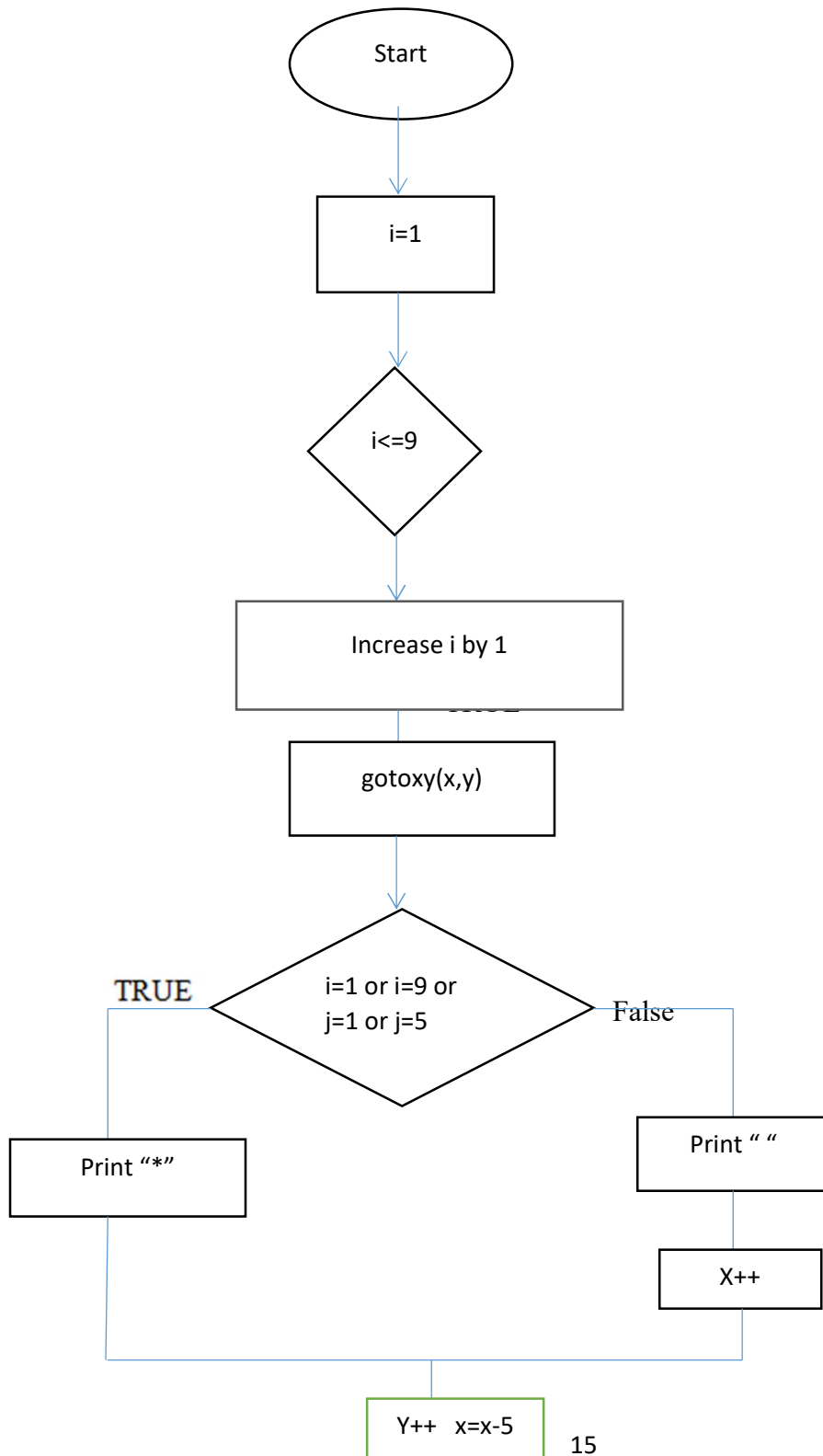
1. Start
2. $i = 1$
3. if $i \leq 9$ increase i by 1
4. then gotoxy(x,y)
5. if $i=1$ OR $i=5$ OR $i=9$ OR $i>5$ and $j=1$ OR $i<5$ and $j=5$ then print *
6. else print “ “
- 7.increase x by 1 and y by 1
8. $x=x-5$ and so on.

F.int main()

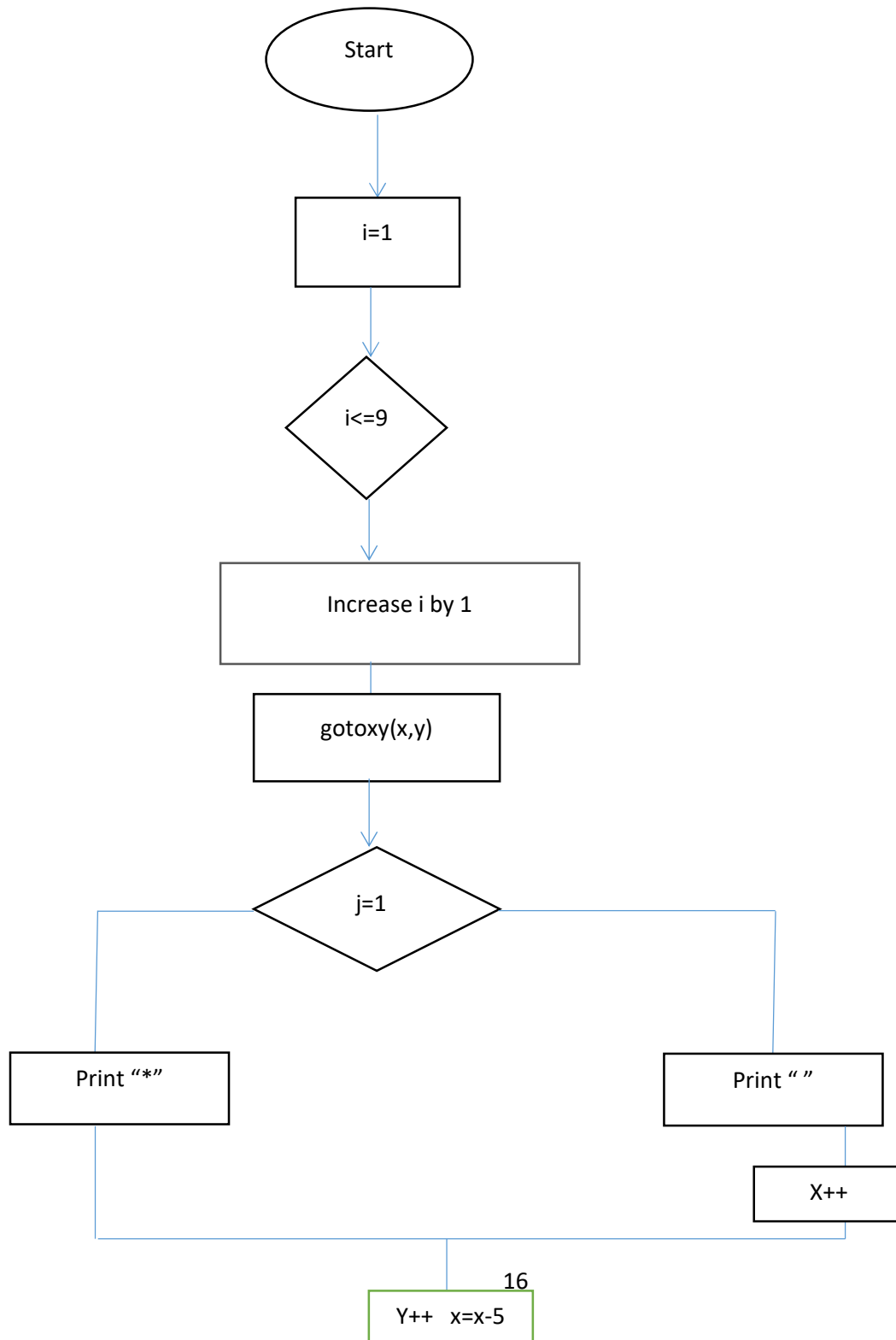
1. Start
 2. input h, m and s
 3. for $h < 24$ increase h by 1
 4. for $m < 60$ increase m by 1
 5. for $s < 60$ increase s by 1
 6. call(h/10,10,7)
 7. colon(30,7)
 8. call (m/10,40,7)
 9. colon (60,7)
 10. call(s/10,70,7)
 11. Sleep(890)
- Call the user defined functions.

Flowchart

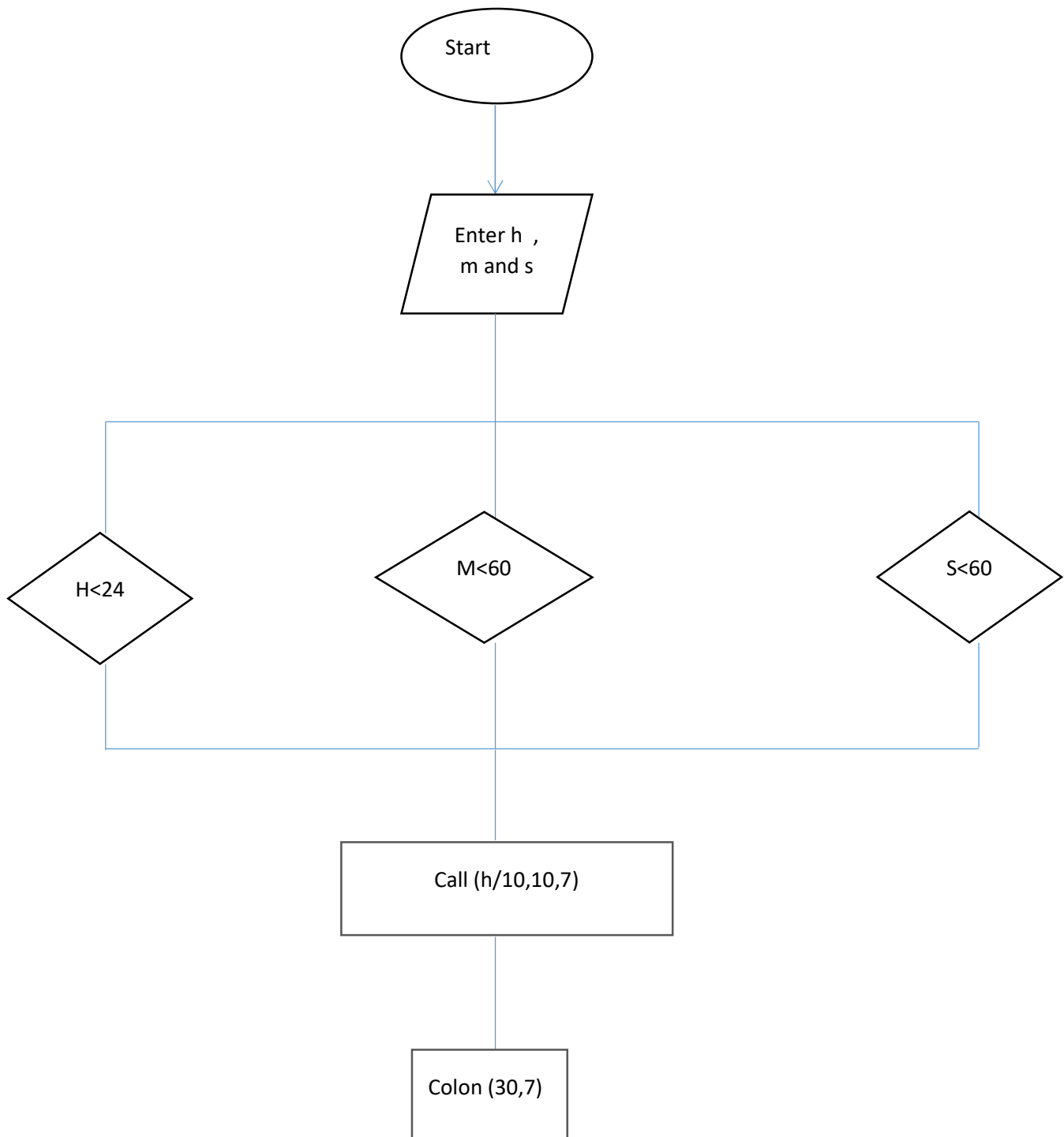
void zero(x,y)



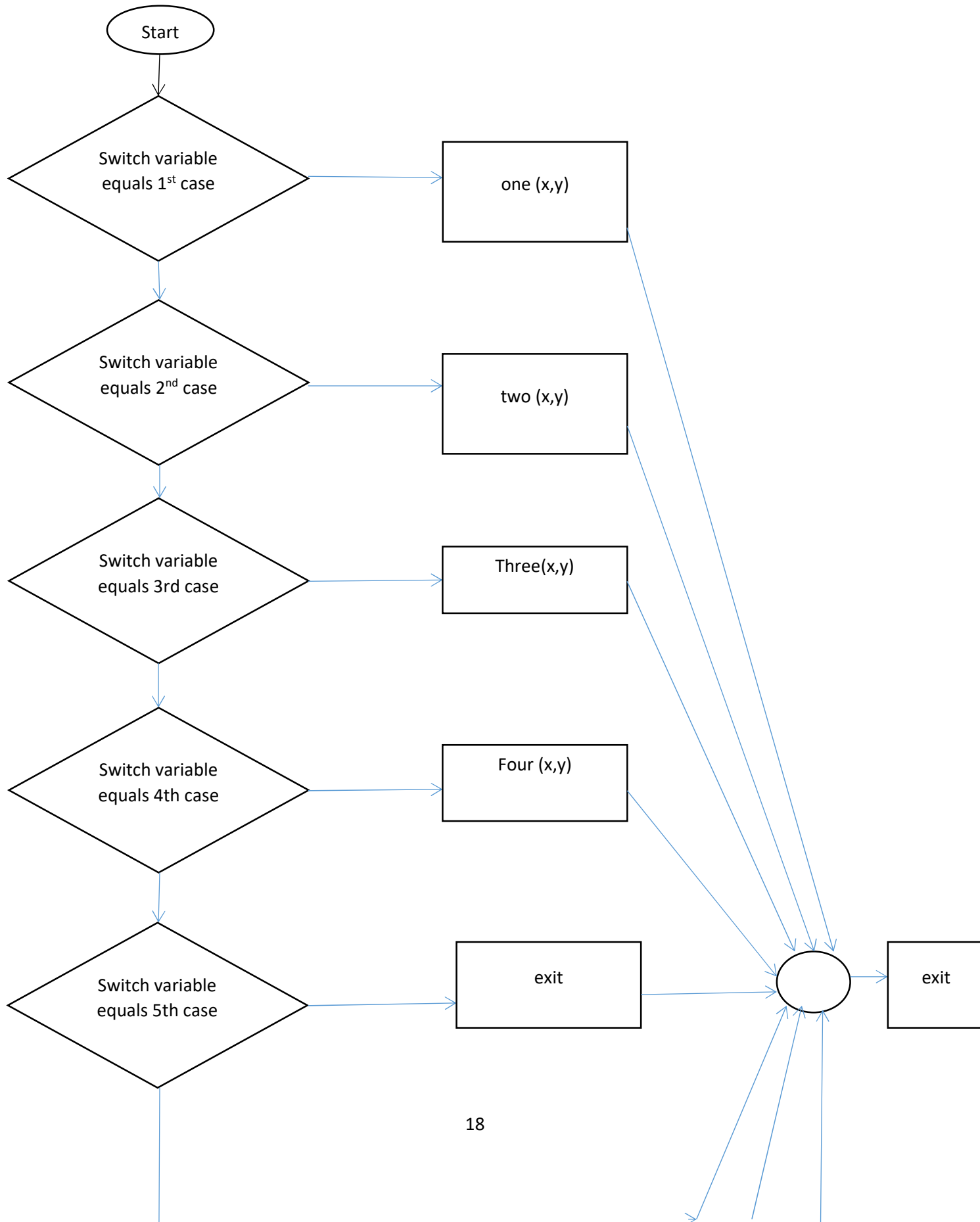
void one(x, v)

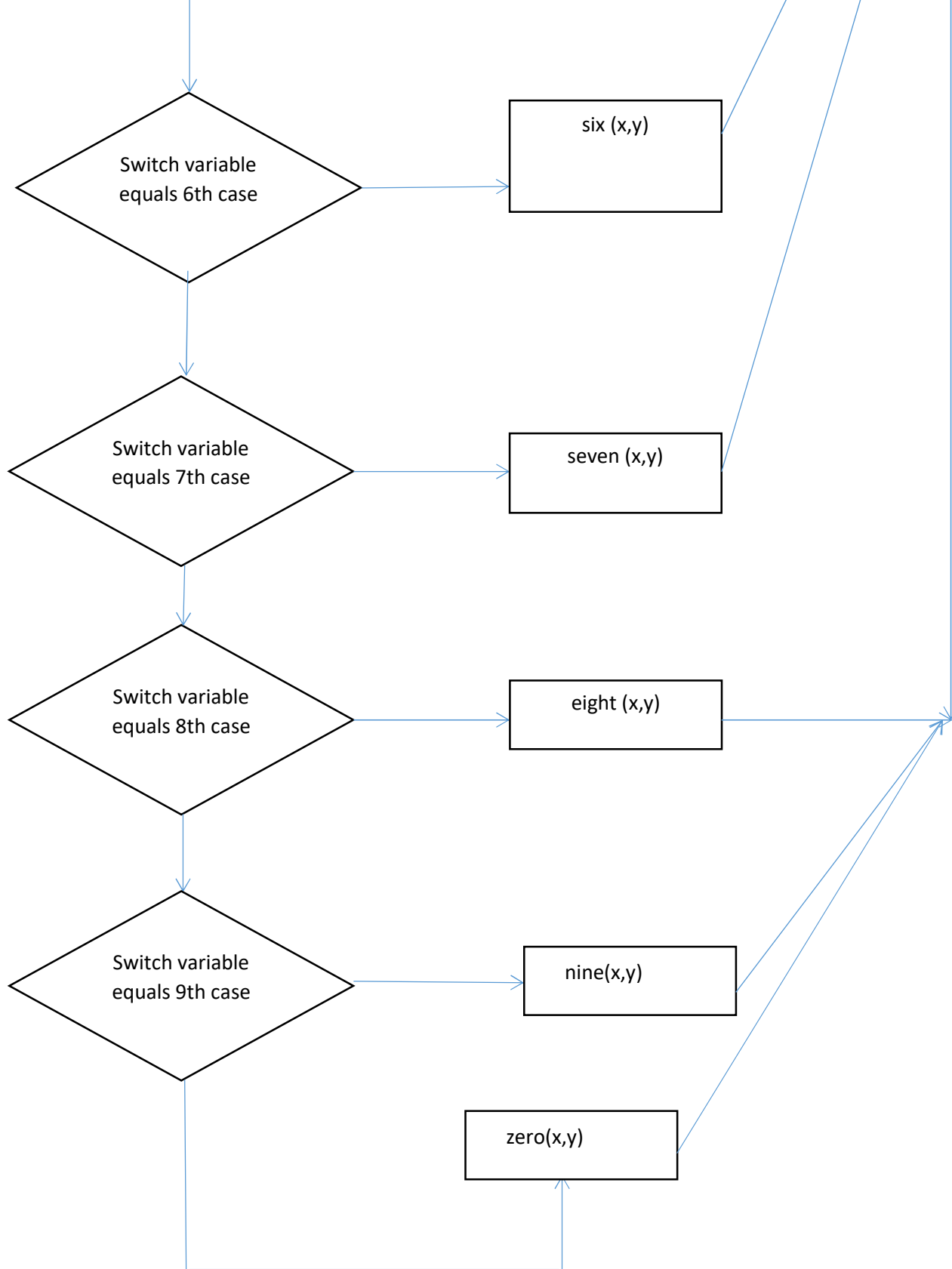


.int main()

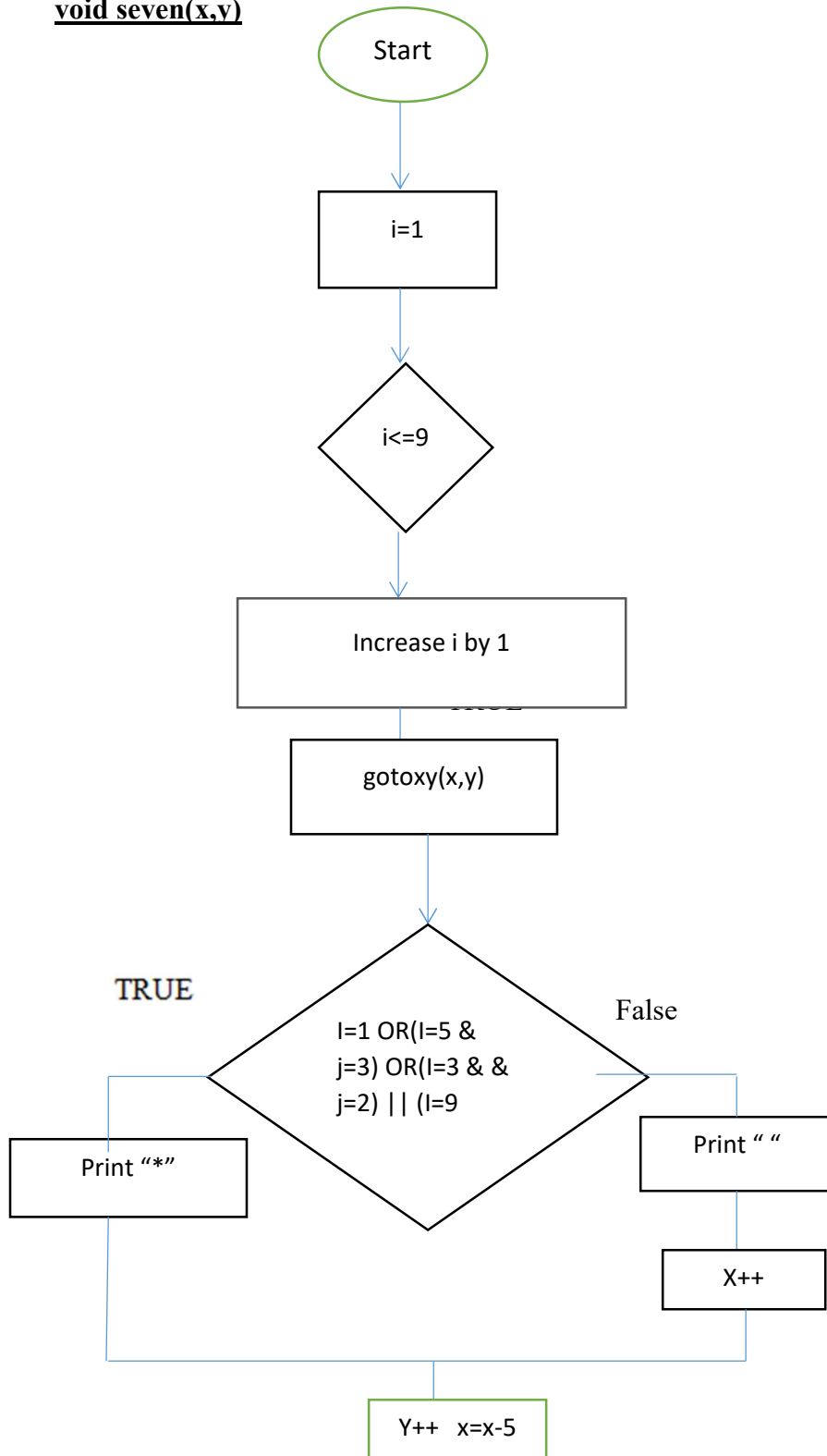


void call (digit, x, y)

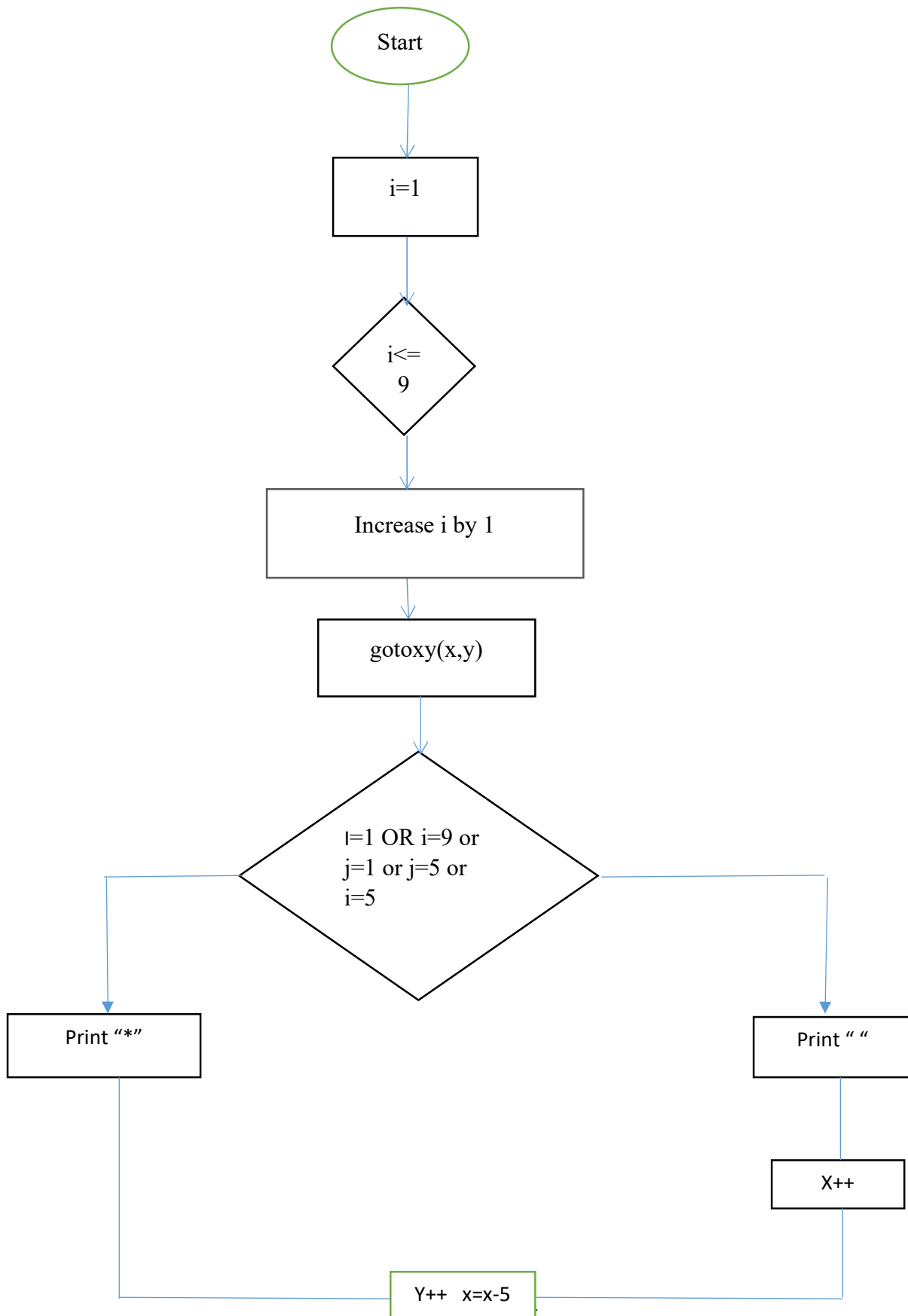




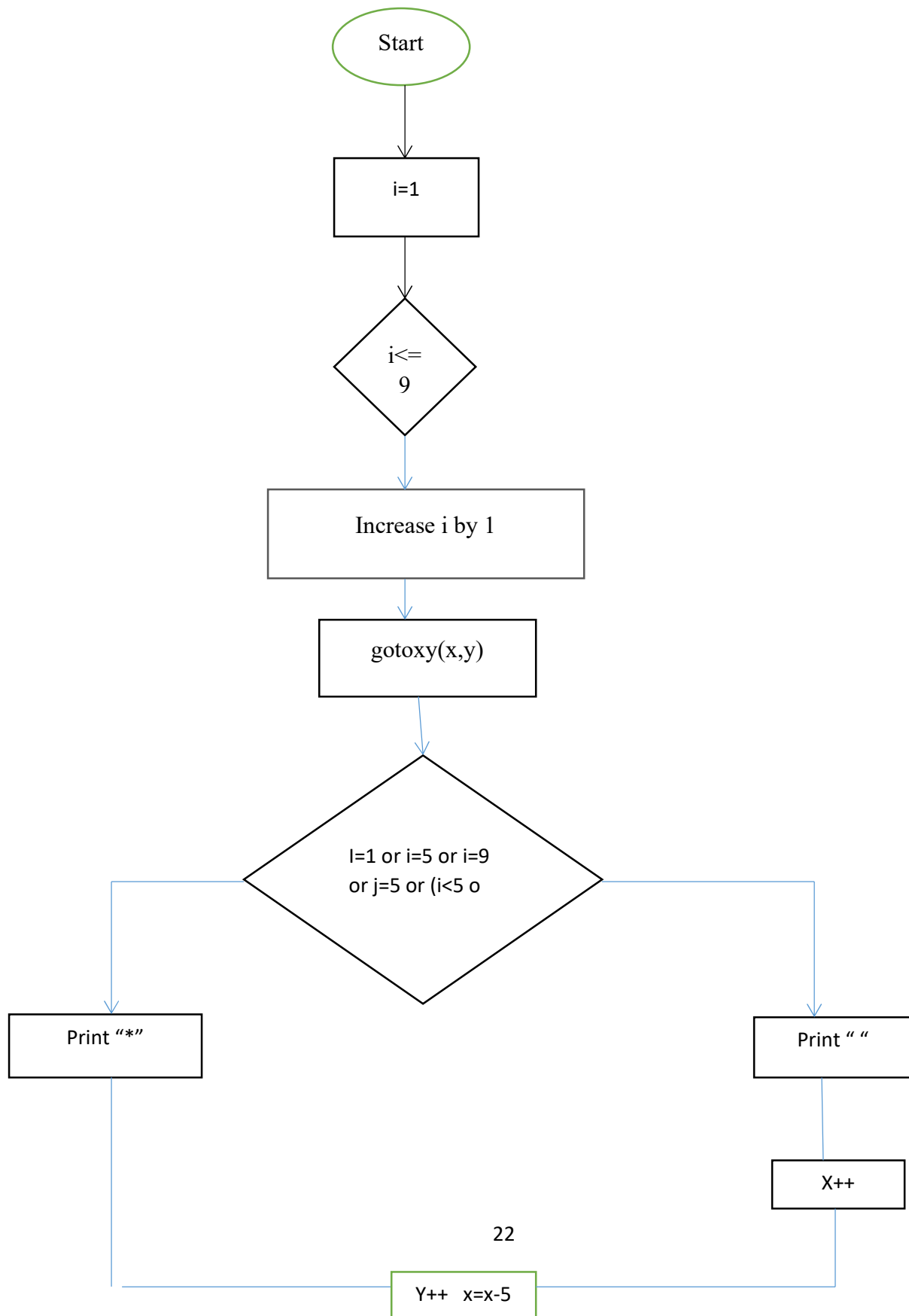
void seven(x,y)



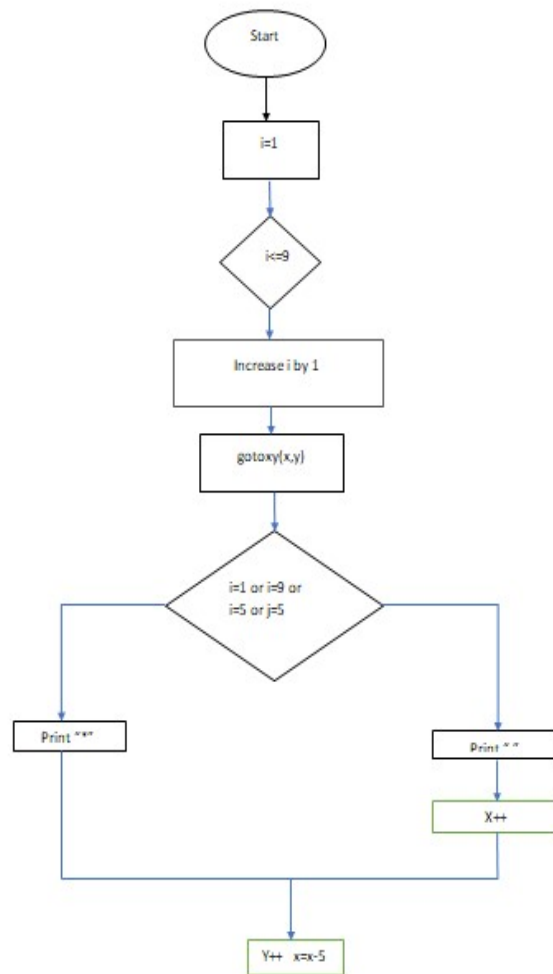
void eight (x,y)



void nine (x,y)

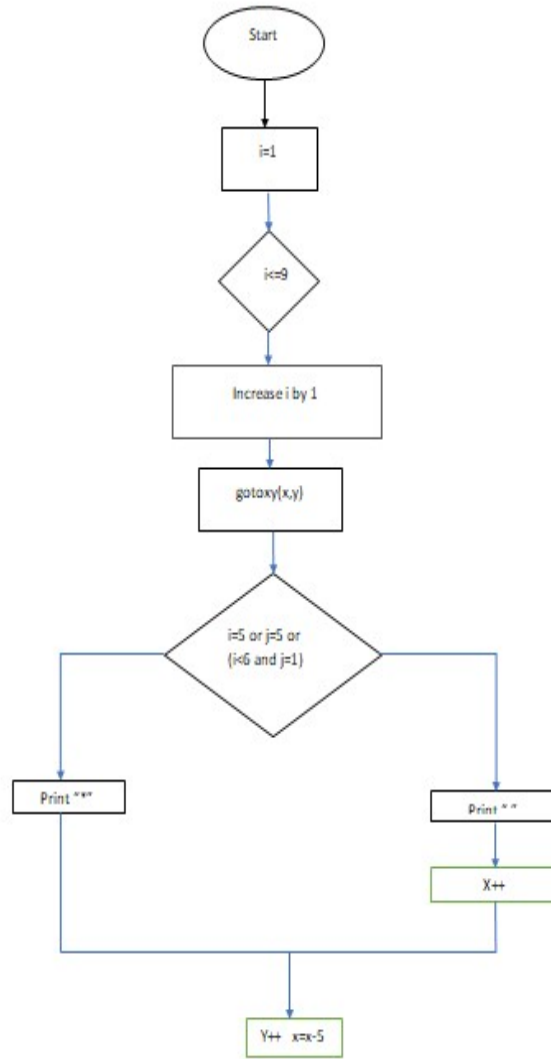


void three(x, y)



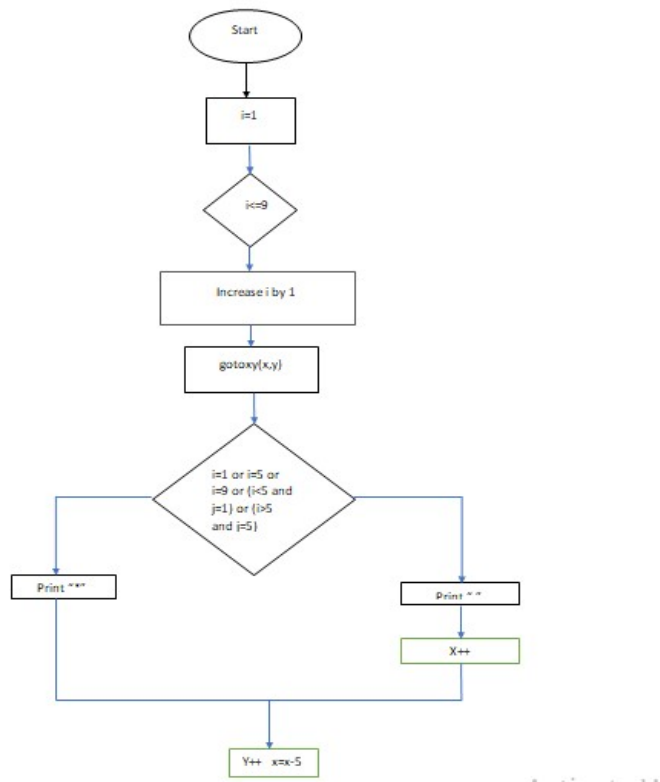
Activate W
Go to Settings

void four(x, y)



Activate
Go to Setti

void five(x, y)



Source Code

```
#include<stdio.h>
#include<windows.h>
#include<string.h>
#include<conio.h>

char ch='*';
struct timezoney
{
    char name[20];
    int hrvalue;
    int minvalue;
};

void zero(int x,int y);
void one(int x,int y);
void two(int x,int y);
void three(int x,int y);
void four(int x,int y);
void five(int x,int y);
void six(int x,int y);
void seven(int x,int y);
void eight(int x,int y);
void nine(int x,int y);
void colon(int x,int y);
void call(int digit,int x,int y);
void gotoxy(int x,int y);
```

```

int main()
{
    int h=0,m=0,s=0;
    char tmzn[20];
    int gmthr;
    int gmtmin;
    int i;
    struct timezoney t[15];
    strcpy(t[0].name,"washingtondc");
    t[0].hrvalue=-4;
    t[0].minvalue=0;
    strcpy(t[1].name,"london");
    t[1].hrvalue=1;
    t[1].minvalue=0;
    strcpy(t[2].name,"kathmandu");
    t[2].hrvalue=5;
    t[2].minvalue=45;
    strcpy(t[3].name,"delhi");
    t[3].hrvalue=5;
    t[3].minvalue=30;
    strcpy(t[4].name,"lisbon");
    t[4].hrvalue=1;
    t[4].minvalue=0;
    strcpy(t[5].name,"tokyo");
    t[5].hrvalue=9;
    t[5].minvalue=0;
    strcpy(t[6].name,"bangkok");
    t[6].hrvalue=7;
    t[6].minvalue=0;

```

```

strcpy(t[7].name,"berlin");
t[7].hrvalue=2;
t[7].minvalue=0;
strcpy(t[8].name,"athens");
t[8].hrvalue=3;
t[8].minvalue=0;
strcpy(t[9].name,"beijing");
t[9].hrvalue=8;
t[9].minvalue=0;
strcpy(t[10].name,"sydney");
t[10].hrvalue=10;
t[10].minvalue=0;
strcpy(t[11].name,"mexico");
t[11].hrvalue=-6;
t[11].minvalue=0;
strcpy(t[12].name,"abudhabi");
t[12].hrvalue=4;
t[12].minvalue=0;

```

```

printf("*****\n");
printf("*** DIGITAL_CLOCK *****\n");
printf("*****\n");

```

```

printf("Programmed by\t Dikshanta Muni Poudel(077bce049)");
printf("\n\t\t Imej Gautam(077bce062)");
printf("\n\t\t Mahesh Khatri(077bce080)");
printf("\nPlease enter time in the format HH:MM:SS\n");

```

```

scanf("%d%d%d",&h,&m,&s);
printf("Please enter your timezone\n");
printf("(You can change timezone later by entering any key)");
scanf("%s",&tmzn);
for(i=0;i<=12;i++)
{
    int temp=strcmp(tmzn,t[i].name);
    if(temp==0)
    {
        break;
    }
}
/*gmthr=h-t[i].hrvalue;
gmtmin=m-t[i].minvalue;*/
goto st;
edit:
printf("Enter the time zone you want");

```

```

char tempn[20];
scanf("%s",&tempn);
int j;
for( j=0;j<=12;j++)
{
    int temp=strcmp(tempn,t[j].name);
    if(temp==0)
    {
        break;
    }
}
h=h+t[j].hrvalue-t[i].hrvalue;

```

```

m=m+t[j].minvalue-t[i].minvalue;

printf("%s,GMT %d:%d",t[j].name,t[j].hrvalue,t[j].minvalue);

goto et;

```

```

st:

printf("%s,GMT %d:%d",t[i].name,t[i].hrvalue,t[i].minvalue);

```

```

et:

```

```

for(h;h<24;h++)
{
    for(m;m<60;m++)
    {
        for(s;s<60;s++)
        {
            call(h/10,31,15);
            call(h%10,41,15);
            colon(48,15);
            call(m/10,55,15);
            call(m%10,65,15);
            colon(73,15);
            call(s/10,80,15);
            call(s%10,88,15);
            Sleep(890);
            if(kbhit()==0)
            {
                continue;
            }
            else
            {

```

```

        goto edit;
    }
}

s=0;
}
m=0;
}
h=0;
goto st;
end:

return 0;
}
void gotoxy(int x,int y)
{
    COORD c;
    c.X=x;
    c.Y=y;
    SetConsoleCursorPosition(GetStdHandle(STD_OUTPUT_HANDLE),c);
}
void call(digit,x,y)
{
    switch(digit)
    {
        case 1: one(x,y); break;
        case 2: two(x,y); break;
        case 3: three(x,y); break;
        case 4: four(x,y); break;
    }
}

```

```

case 5: five(x,y); break;
case 6: six(x,y); break;
case 7: seven(x,y); break;
case 8: eight(x,y); break;
case 9: nine(x,y); break;
default: zero(x,y); break;
}
}
void zero(x,y)
{
for(int I=1;I<=9;I++)
{
for(int j=1;j<=5;j++)
{
gotoxy(x,y);
if(I==1 || I==9 || j==1 || j==5)
{printf("%c",ch);}
else
{printf(" ");}
x++;
}
y++;
x=x-5;
printf("\n");
}
}

void one(x,y)
{for(int I=1;I<=9;I++)
{

```



```

for(int j=1;j<=5;j++)
{
    gotoxy(x,y);
    if(j==1)
    {printf("%c",ch);}
    else
    {printf(" ");}
    x++;
}
y++;
x=x-5;
printf("\n");
}
}

```

```

void two(x,y)
{
    for(int I=1;I<=9;I++)
    {
        for(int j=1;j<=5;j++)
        {
            gotoxy(x,y);
            if(I==1 || I==5 || I==9 || (I>5 & j==1 ) || (I<5 & j==5))
            {printf("%c",ch);}
            else
            {printf(" ");}
            x++;
        }
        y++;
        x=x-5;
    }
}

```

```

printf("\n");
}}
void three(x,y)
{for(int I=1;I<=9;I++)
{
for(int j=1;j<=5;j++)
{
gotoxy(x,y);
if(I==1 ||I==9 ||j==5 ||I==5)
{printf("%c",ch);}
else
{printf(" ");}
x++;
}
y++;
x=x-5;
printf("\n");
}
}

```

```

void four(x,y)
{for(int I=1;I<=9;I++)
{
for(int j=1;j<=5;j++)
{
gotoxy(x,y);
if(I==5 ||j==5 ||(I<6 &j==1))
{printf("%c",ch);}
else
{printf(" ");}

```

```

x++;
}
y++;
x=x-5;
printf("\n");
}}

```

```

void five(x,y)
{
for(int I=1;I<=9;I++)
{
for(int j=1;j<=5;j++)
{
gotoxy(x,y);
if(I==1 || I==5 || I==9 || (I<5 & j==1 ) || (I>5 & j==5))
{printf("%c",ch);}
else
{printf(" ");}
x++;
}
y++;
x=x-5;
printf("\n");
}}

```

```

void six(x,y)
{
for(int I=1;I<=9;I++)
{
for(int j=1;j<=5;j++)

```

```

{
gotoxy(x,y);
if(I==1 || I==5 || I==9 || j==1 || (I>5 & j==5))
{printf("%c",ch);}
else
{printf(" ");}
x++;
}
y++;
x=x-5;
printf("\n");
}}

void seven(x,y)
{
for(int I=1;I<=9;I++)
{
for(int j=1;j<=5;j++)
{
gotoxy(x,y);
if(I==1 || (I==5 & j==3) || (I==3 & j==4) || (I==7 & j==2) || (I==9 & j==1) )
{

printf(" ");}
x++;
}
y++;
x=x-5;
printf("\n");
}}

```

```

void eight(x,y)
{

for(int I=1;I<=9;I++)
{
for(int j=1;j<=5;j++)
{
gotoxy(x,y);
if(I==1 || I==9 || j==1 || j==5 || I==5)
{printf("%c",ch);}
else
{printf(" ");}
x++;
}
y++;
x=x-5;
printf("\n");
}
}

```

```

void nine(int x,int y)
{
for(int I=1;I<=9;I++)
{
for(int j=1;j<=5;j++)
{
gotoxy(x,y);
if(I==1 || I==5 || I==9 || j==5 || (I<5 & j==1))

```

```
{printf("%c",ch);}
```

```
else
```

```
{printf(" ");}
```

```
x++;
```

```
}
```

```
y++;
```

```
x=x-5;
```

```
printf("\n");
```

```
}
```

```
}
```

```
void colon(x,y)
```

```
{
```

```
for(int I=1;I<=9;I++)
```

```
{
```

```
for(int j=1;j<=5;j++)
```

```
{
```

```
gotoxy(x,y);
```

```
if((I==3 &j==3 )||(I==7 &j==3))
```

```
{printf("*");}
```

```
else
```

```
{printf(" ");}
```

```
x++;
```

```
}
```

```
y++;
```

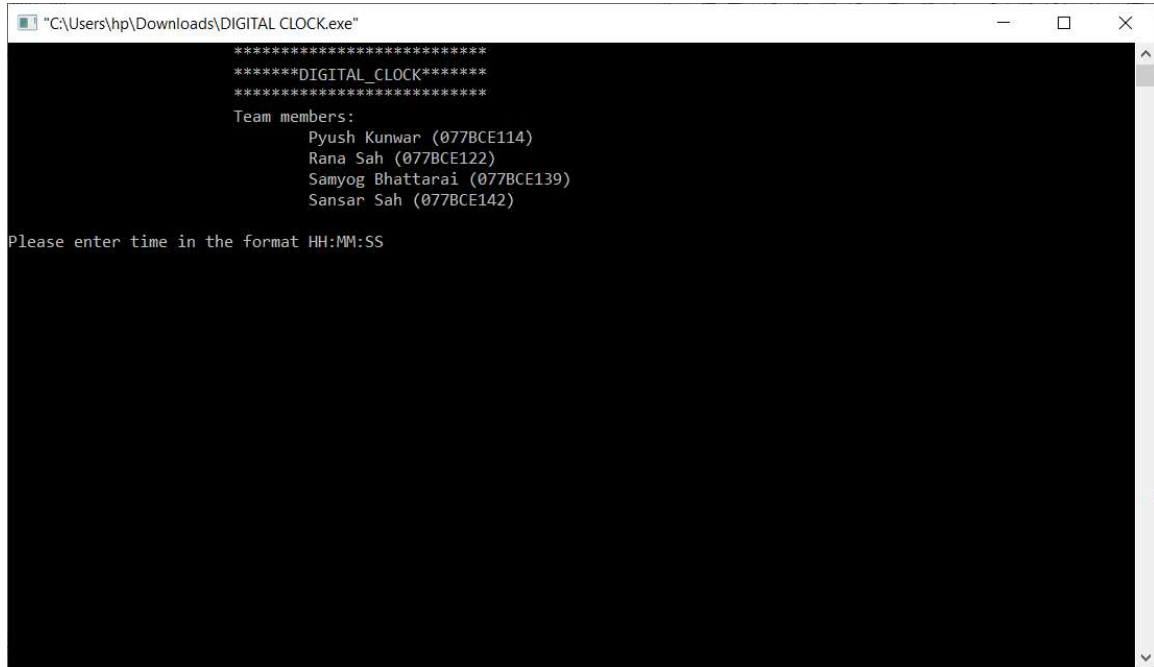
```
x=x-5;
```

```
printf("\n");
```

```
}
```

```
}
```

Output

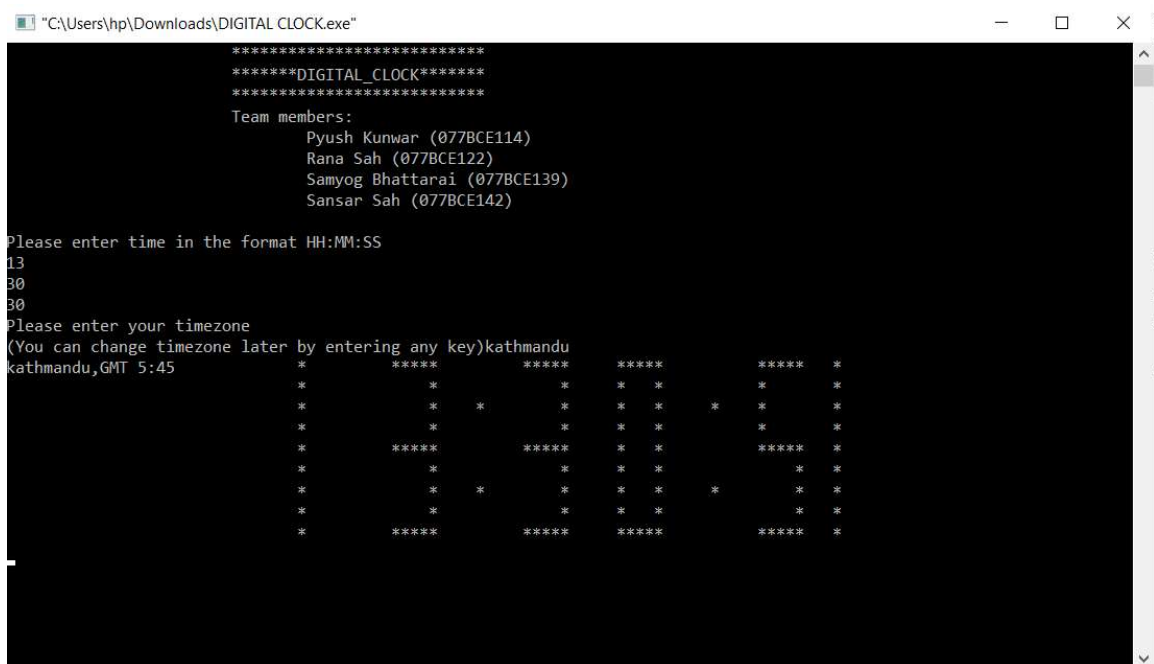


```
"C:\Users\hp\Downloads\DIGITAL CLOCK.exe"

*****
*****DIGITAL_CLOCK*****
*****
Team members:
    Pyush Kunwar (077BCE114)
    Rana Sah (077BCE122)
    Samyog Bhattarai (077BCE139)
    Sansar Sah (077BCE142)

Please enter time in the format HH:MM:SS
```

Fig: Console screen of output



```
"C:\Users\hp\Downloads\DIGITAL CLOCK.exe"

*****
*****DIGITAL_CLOCK*****
*****
Team members:
    Pyush Kunwar (077BCE114)
    Rana Sah (077BCE122)
    Samyog Bhattarai (077BCE139)
    Sansar Sah (077BCE142)

Please enter time in the format HH:MM:SS
13
30
30
Please enter your timezone
(You can change timezone later by entering any key)kathmandu
kathmandu,GMT 5:45

*      *      *      *      *      *
*      *      *      *      *      *
*      *      *      *      *      *
*      *      *      *      *      *
*      *      *      *      *      *
*      *      *      *      *      *
*      *      *      *      *      *
*      *      *      *      *      *
*      *      *      *      *      *
*      *      *      *      *      *
*      *      *      *      *      *
```

Fig: Digital clock

Conclusion

After the testing and running of this program following conclusions can be made:

1. This program enables to understand the concepts of loop and switch statements
2. This program also allows to learn how a digital clock works.
3. This program also shows how various techniques can be used to make the program visually appealing.