

CSE1310 INLAB 06

Note: Use the file you created in your previous lab (inlab5) checkpoint 7 for the first checkpoint.

Determine the highest and the lowest score obtained among the 5 test scores entered. You can use either multiple 'if-else' selection control structures with the help of both relational and binary logical operators in the conditional statements. After you have done so, signal the grader.

Checkpoint 1

Note: Create a new directory by name “lab6”. All the files for inlab 6 should be saved in this directory only. Names of the files should follow the format ‘insurance-chk1.c’, ‘insurance-chk2.c’ etc. After you complete a checkpoint, make a copy of that file for the next checkpoint and modify the copied file.

Open a file named ‘**insurance-chk1.c**’ and type the following code:

```
/* insurance.c
This is a program to determine if a person qualifies for a discount in the insurance
premium of a vehicle. */
#include <stdio.h>
int main(void )
{
    char gender, marital_status; /* Variables to accept gender and marital_status */
    int age; /* Variable to accept age of the person */
    printf(“Enter the age, marital status and gender of the person: \n”);
    scanf(“%d %c %c”, &age, &marital_status, &gender);
    if ((marital_status = 'A') || (marital_status = 'a'))
    {
        printf(“The person is eligible for a discount.\n”);
    }
    else
    {
        if ((marital_status = 'S') || (marital_status = 's'))
        {
            if (age > 30)
                printf(“The person is eligible for a discount.\n”);
            else
                printf(“The person is not eligible for a discount.\n”);
        }
        else
            printf(“Invalid entry.\n”);
    }
    return (0);
}
```

```
}
```

Save the program and exit 'vi'. Compile the program. Fix the logical and compilation errors (hint: comparison operator) that you encounter and recompile the program. Signal the grader at this point.

Checkpoint 2

Modify the above program using multiple-alternative decision structures along with logical operators to guarantee that a discount in the insurance is provided to a person under the following conditions:

- If the person is married, 'male', and 25 or above years of age
- If the person is married, 'female', and 21 or above years of age

Use 'M' or 'm' to represent 'male' and 'F' or 'f' to represent 'female'. If the user enters any other character in place of these four, the program should print 'Invalid entry' and quit.

This can be done by putting the following code within the first 'if' statement:

```
if (gender == 'M' || gender == 'm')
{
    if(age >= 25)
        /* print that person is eligible */
    else
        /* print that person is not eligible */
}
else if( gender == 'F' || gender == 'f')
{
    if(age >= 21)
        /* print that person is eligible */
    else
        /* print that person is not eligible */
}
else
{
    /* print that entry is invalid */
}
```

Substitute appropriate code for the comments in the above example. Also, validate the input for age to make sure that it is a positive integer. In case of an invalid input, ask the user to enter a valid input value for age. This validation should take place after the inputs are read from the standard input. Thus the code should be changed accordingly.

For the case when marital status is 'Single', do not make any changes. Your code should be added within the existing if-else statements, in the appropriate places. Signal the grader when you complete this section. It is necessary to display results for the different cases that exist.

Checkpoint 3

Include a condition for the gender criteria when a person is 'Single'. This part should display proper results for the following condition:

- If the person is 'Single' and 21 or above years of age and either 'male' or 'female' Use relational and/or logical operators in your 'if' conditions. Compile the program using 'gcc'. If the program compiles without any errors, execute the program using the 'a.out' executable to get the output. If the program has compilation errors, you need to fix them before executing the program. Signal the grader when you finish.

Checkpoint 4

Selective Structure: *switch*.

The syntax of the *switch* instruction is a bit peculiar. Its objective is to check several possible constant values for an expression, something similar to several *if* and *else if* sentences. Its form is the following:

```
switch (expression) {  
  case constant1:  
    block of instructions 1  
  break;  
  case constant2:  
    block of instructions 2  
  break;  
  .  
  .  
  .  
  default:  
    default block of instructions  
}
```

An example conversion of *if* and *else if* control structure to *switch* selective structure:

```
if ( code == '+' )  
    result = x + y;  
else if ( code == '-' )  
    result = x - y;  
else if ( code == '*' )  
    result = x * y;  
else if ( code == '/' )  
    result = x / y;  
else  
    printf ( "Operator invalid!" );
```

```

switch ( code )
{
    case '+' :
        result = x + y;
        break;
    case '-' :
        result = x - y;
        break;
    case '*' :
        result = x * y;
        break;
    case '/' :
        result = x / y;
        break;
    default :
        printf ( "Operator invalid!" );
        break;
}

```

Modify the above program of checkpoint 4 to use 'switch' statement with 'marital_status' being the switch condition. The rest of the program will follow the same logic as before. Use if-else statements inside your case labels. Here is the outline of the switch statement:

```

switch(marital_status)
{
    case 'A':
    case 'a':
        /* Use existing code here to find out if person is eligible for
        discount */
        break;
    case 'S':
    case 's':
        /* Use existing code here to find out if person is eligible for
        discount */
        break;
    default:
        /* print that entry is invalid */
}

```

When you've completed this, compile and execute the program as before and with the output on the screen, signal the grader to check your work.

Checkpoint 5

Now remove *break* statement from the first case. Compile and run the program. Observe the output when you provide input for *marital_status* as 'A' or 'a' signal the grader to

check your work and tell lab instructor your observation. Put back the *break* statement that you just removed.

Checkpoint 6