**SOFTWARE TESTING LABORATORY**

**Subject Code: 21ISL66 I.AMarks:50**

**Hours/Week: 02 Exam Hours: 03**

**Total Hours: 24 Exam Marks: 50 Number of Lecture Hours/Week: 02P**

**PART A – List of problems for which student should develop program and execute in theLaboratory**

1. Design, develop, code and run the program in any suitable language to solve the commission problem. Analyze it from the perspective of boundary value testing, derive different test cases, execute these test cases and discuss the test results.
2. Design, develop, code and run the program in any suitable language to implement the NextDate function. Analyze it from the perspective of equivalence class value testing, derive different test cases, execute these test cases and discuss the test results.
3. Design, develop, code and run the program in any suitable language to solve the commission problem. Analyze it from the perspective of decision table-based testing, derive different test cases, execute these test cases and discuss the test results.
4. Design and develop a program in a language of your choice to solve the triangle problem defined as follows: Accept three integers which are supposed to be the three sides of a triangle and determine if the three values represent an equilateral triangle, isosceles triangle, scalene triangle, or they do not form a triangle at all. Assume that the upper limit for the size of any side is 10. Derive test cases for your program based on boundary-value analysis, equivalence class partitioning and decision-table approach and execute the test cases and discuss the results.
5. Design, develop, code and run the program in any suitable language to solve the commission problem. Analyze it from the perspective of dataflow testing, derive different test cases, execute these test cases and discuss the test results.
6. Design, develop, code and run the program in any suitable language to implement the binary search algorithm. Determine the basis paths and using them derive different test cases, execute these test cases and discuss the test results

**PART B – Practical Based Learning**

01 Develop a Mini Project with documentation of suitable test-cases and their results to perform automation testing of anyE-commerce or social media web page. Suggested Guidelines:

● Create a WebDriver session.

● Navigate to a Web page.

● Locate the web elements on the navigated page.

● Perform an actions on the located elements.

● Assert the performed actions did the correct thing.

● Report the results of the assertions.

● End the session.

Each inputs / data feeds (ex: website, username, password, mobile no, product name, etc.,)must be provided through a file linked with code and neither to be entered manually nor to be included in the code Use any software testing tool like selenium, Katalon, etc.,

1. Design, develop, code and run the program in any suitable language to solve the commission problem. Analyze it from the perspective of boundary value testing, derive different test cases, execute these test cases and discuss the test results.

ALGORITHM

STEP 1: Define lockPrice=45.0, stockPrice=30.0, barrelPrice=25.0

STEP2: Input locks

STEP3: while(locks!=-1) ‘input device uses -1 to indicate end of data

goto STEP 12

STEP4: input (stocks, barrels)

STEP5: compute lockSales, stockSales, barrelSales and sales

STEP6: output(“Total sales:” sales)

STEP7: if (sales > 1800.0) goto STEP 8 else goto STEP 9

STEP8: commission=0.10\*1000.0; commission=commission+0.15 \* 800.0;

commission = commission + 0.20 \* (sales-1800.0)

STEP9: if (sales > 1000.0) goto STEP 10 else goto STEP 11

STEP10: commission=0.10\* 1000.0; commission=commission + 0.15 \*

(sales-1000.0)

STEP11: Output(“Commission is $”, commission)

STEP12: exit

PROGRAM CODE:

#include <stdio.h>

int main() {

int locks, stocks, barrels, tlocks, tstocks, tbarrels;

float lprice, sprice, bprice, sal es, comm;

int c1, c2, c3, temp;

lprice = 45.0;

sprice = 30.0;

bprice = 25.0;

tlocks = 0;

tbarrels = 0;

tstocks = 0;

printf("Enter the number of locks and to exit the loop enter -1 for locks\n");

scanf("%d", &locks);

while (locks != -1) {

c1 = (locks <= 0) || (locks > 70);

printf("Enter the number of stocks and barrels: ");

scanf("%d%d", &stocks, &barrels);

c2 = (stocks <= 0 || stocks > 80);

c3 = (barrels <= 0 || barrels > 90);

if (c1)

printf("Value of locks not in the range 1-70\n");

else {

temp = tlocks + locks;

if (temp > 70)

printf("New total locks = %d not in the range 1-70\n", temp);

else

tlocks = temp;

}

printf("Total locks = %d\n", tlocks);

if (c2)

printf("Value of stocks not in the range 1-80\n");

else {

temp = tstocks + stocks;

if (temp > 80)

printf("New total stocks = %d not in the range 1-80\n", temp);

else

tstocks = temp;

}

printf("Total stocks = %d\n", tstocks);

if (c3)

printf("Value of barrels not in the range 1-90\n");

else {

temp = tbarrels + barrels;

if (temp > 90)

printf("New total barrels = %d not in the range 1-90\n", temp);

else

tbarrels = temp;

}

printf("Total barrels = %d\n", tbarrels);

printf("Enter the number of locks and to exit the loop enter -1 for locks\n");

scanf("%d", &locks);

}

printf("Total locks = %d\nTotal stocks = %d\nTotal barrels = %d\n", tlocks, tstocks, tbarrels);

sales = lprice \* tlocks + sprice \* tstocks + bprice \* tbarrels;

printf("Total sales = %f\n", sales);

if (sales > 0) {

if (sales > 1800.0) {

comm = 0.10 \* 1000;

comm = comm + 0.15 \* 800;

comm = comm + (0.20 \* (sales - 1800));

} else if (sales > 1000) {

comm = 0.10 \* 1000;

comm = comm + 0.15 \* (sales - 1000);

} else {

comm = 0.10 \* sales; // Fixed the commission for sales <= 1000

}

printf("The total commission is %f\n", comm);

} else {

printf("There is no sales\n");

}

return 0;

}

**Report:**

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Case Id** | **Description** | **Input Data** | |  | **Expected Output** | **Actual Output** | **Comments** |
| **a** | **b** | **c** |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

2.Design, develop, code and run the program in any suitable language to implement the NextDate function. Analyze it from the perspective of equivalence class value testing, derive different test cases, execute these test cases and discuss the test results.

ALGORITHM

STEP 1: Input date in format DD.MM.YYYY

STEP2: if MM is 01, 03, 05,07,08,10 do STEP3 else STEP6

STEP3:if DD < 31 then do STEP4 else if DD=31 do STEP5 else

output(Invalid Date);

STEP4: tomorrowday=DD+1 goto STEP18

STEP5: tomorrowday=1; tomorrowmonth=month + 1 goto STEP18

STEP6: if MM is 04, 06, 09, 11 do STEP7

STEP7: if DD<30 then do STEP4 else if DD=30 do STEP5 else

output(Invalid Date);

STEP8: if MM is 12

STEP9: if DD<31 then STEP4 else STEP10

STEP10: tomorrowday=1, tommorowmonth=1, tommorowyear=YYYY+1;

goto STEP18

STEP11: if MM is 2

STEP12: if DD<28 do STEP4 else do STEP13

STEP13: if DD=28 & YYYY is a leap do STEP14 else STEP15

STEP14: tommorowday=29 goto STEP18

STEP15: tommorowday=1, tomorrowmonth=3, goto STEP18;

STEP16: if DD=29 then do STEP15 else STEP17

STEP17: output(“Cannot have feb”, DD); STEP19

STEP18: output(tomorrowday, tomorrowmonth, tomorrowyear);

STEP19: exit

PROGRAM CODE:

#include <stdio.h>

int check(int day, int month) {

if ((month == 4 || month == 6 || month == 9 || month == 11) && day == 31) {

return 1;

} else {

return 0;

}

}

int isleap(int year) {

if ((year % 4 == 0 && year % 100 != 0) || (year % 400 == 0)) {

return 1;

} else {

return 0;

}

}

int main() {

int day, month, year, tomm\_day, tomm\_month, tomm\_year;

char flag;

do {

flag = 'y';

printf("\nEnter today's date in the form of dd mm yyyy: \n");

scanf("%d %d %d", &day, &month, &year);

tomm\_month = month;

tomm\_year = year;

if (day < 1 || day > 31) {

printf("Value of the day not in the range 1…31\n");

flag = 'n';

}

if (month < 1 || month > 12) {

printf("Value of the month not in the range 1---12\n");

flag = 'n';

} else if (check(day, month)) {

printf("The value of the day not in the range day <= 30\n");

flag = 'n';

}

if (year <= 1812 || year > 2018) {

printf("The value of the year not in the range 1812…2018\n");

flag = 'n';

}

if (month == 2) {

if (isleap(year) && day > 29) {

printf("Invalid date input for leap year\n");

flag = 'n';

} else if (!(isleap(year)) && day > 28) {

printf("Invalid date input for not a leap year\n");

flag = 'n';

}

}

} while (flag == 'n');

switch (month) {

case 1:

case 3:

case 5:

case 7:

case 8:

case 10:

if (day < 31) {

tomm\_day = day + 1;

} else {

tomm\_day = 1;

tomm\_month = month + 1;

}

break;

case 4:

case 6:

case 9:

case 11:

if (day < 30) {

tomm\_day = day + 1;

} else {

tomm\_day = 1;

tomm\_month = month + 1;

}

break;

case 12:

if (day < 31) {

tomm\_day = day + 1;

} else {

tomm\_day = 1;

tomm\_month = 1;

if (year == 2018) {

printf("The next day is out of boundary value of year\n");

}

tomm\_year = year + 1;

}

break;

case 2:

if (day < 28) {

tomm\_day = day + 1;

} else if (isleap(year) && day == 28) {

tomm\_day = day + 1;

} else if (day == 28 || day == 29) {

tomm\_day = 1;

tomm\_month = 3;

}

break;

}

printf("Next day is %02d %02d %04d\n", tomm\_day, tomm\_month, tomm\_year);

return 0;

}

Test Report:

3. Design, develop, code and run the program in any suitable language to solve the commission problem. Analyze it from the perspective of decision table-based testing, derive different test cases, execute these test cases and discuss the test results.

ALGORITHM

Step 1: Input 3 integer numbers which represents number of Locks, Stocks

and Barrels sold.

Step 2: compute the total sales =(Number of Locks sold \*45) + (Number of

Stocks sold \*30) + (Number of Barrels sold \*25)

Step 3: if a totals sale in dollars is less than or equal to $1000

then commission = 0.10\* total Sales do step 6

Step 4: else if total sale is less than $1800

then commission1 = 0.10\* 1000

commission = commission1 + (0.15 \* (total sales – 1000))

do step 6

Step 5: else commission1 = 0.10\* 1000

commission2 = commission1 + (0.15 \* 800))

commission = commission2 + (0.20 \* (total sales – 1800)) do

step 6

Step 6: Print commission.

Step 7: Stop.

PROGRAM CODE:

#include <stdio.h>

#include <conio.h>

int main() {

int locks, stocks, barrels, tlocks, tstocks, tbarrels;

float lprice, sprice, bprice, sales, comm;

int c1, c2, c3, temp;

lprice = 45.0;

sprice = 30.0;

bprice = 25.0;

tlocks = 0;

tbarrels = 0;

tstocks = 0;

printf("Enter the number of locks and to exit the loop enter -1 for locks\n");

scanf("%d", &locks);

while (locks != -1) {

c1 = (locks <= 0) || (locks > 70);

printf("Enter the number of stocks and barrels: ");

scanf("%d%d", &stocks, &barrels);

c2 = (stocks <= 0 || stocks > 80);

c3 = (barrels <= 0 || barrels > 90);

if (c1)

printf("Value of locks not in the range 1----70\n");

else {

temp = tlocks + locks;

if (temp > 70)

printf("New total locks = %d not in the range 1---70\n", temp);

else

tlocks = temp;

}

printf("Total locks = %d\n", tlocks);

if (c2)

printf("Value of stocks not in the range 1----80\n");

else {

temp = tstocks + stocks;

if (temp > 80)

printf("New total stocks = %d not in the range 1---80\n", temp);

else

tstocks = temp;

}

printf("Total stocks = %d\n", tstocks);

if (c3)

printf("Value of barrels not in the range 1----90\n");

else {

temp = tbarrels + barrels;

if (temp > 90)

printf("New total barrels = %d not in the range 1---90\n", temp);

else

tbarrels = temp;

}

printf("Total barrels = %d\n", tbarrels);

printf("Enter the number of locks & to exit the loop enter -1 for locks\n");

scanf("%d", &locks);

}

printf("Total locks = %d\nTotal stocks = %d\nTotal barrels = %d\n", tlocks, tstocks, tbarrels);

sales = lprice \* tlocks + sprice \* tstocks + bprice \* tbarrels;

printf("\nTotal sales = %f\n", sales);

if (sales > 0) {

if (sales > 1800.0) {

comm = 0.10 \* 1000;

comm = comm + 0.15 \* 800;

comm = comm + (0.20 \* (sales - 1800));

} else if (sales > 1000) {

comm = 0.10 \* 1000;

comm = comm + 0.15 \* (sales - 1000);

} else {

comm = 0.10 \* sales; // Correct calculation for sales <= 1000

}

printf("The total commission is %f", comm);

} else {

printf("There is no sales\n");

}

return 0;

}

Test Report:

|  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **RULES** |  | **R1** | **R2** | **R3** | **R4** | **R5** | **R6** | **R7** | **R8** | **R9** | **R10** | **R11** |
| **Conditions** | C1: |  |  |  |  |  |  |  |  |  |  |  |
| C2 : |  |  |  |  |  |  |  |  |  |  |  |
| C3 : |  |  |  |  |  |  |  |  |  |  |  |
| C4 : |  |  |  |  |  |  |  |  |  |  |  |
| C5 : |  |  |  |  |  |  |  |  |  |  |  |
| C6 : |  |  |  |  |  |  |  |  |  |  |  |
| **Actions** | a1 : |  |  |  |  |  |  |  |  |  |  |  |
| a2 : |  |  |  |  |  |  |  |  |  |  |  |
| a3 : |  |  |  |  |  |  |  |  |  |  |  |
| a4 : |  |  |  |  |  |  |  |  |  |  |  |
| a5 : |  |  |  |  |  |  |  |  |  |  |  |

4. Design and develop a program in a language of your choice to solve the triangle problem defined as follows: Accept three integers which are supposed to be the three sides of a triangle and determine if the three values represent an equilateral triangle, isosceles triangle, scalene triangle, or they do not form a triangle at all. Assume that the upper limit for the size of any side is 10. Derive test cases for your program based on boundary-value analysis, equivalence class partitioning and decision-table approach and execute the test cases and discuss the results.

ALGORITHM:

Step 1: Input a, b & c i.e three integer values which represent three sides of

the triangle.

Step 2: if (a < (b + c)) and (b < (a + c)) and (c < (a + b) then

do step 3

else

print not a triangle. do step 6.

Step 3: if (a=b) and (b=c) then

Print triangle formed is equilateral. do step 6.

Step 4: if (a ≠ b) and (a ≠ c) and (b ≠ c) then

Print triangle formed is scalene. do step 6.

Step 5: Print triangle formed is Isosceles.

Step 6: stop

PROGRAM CODE

#include <stdio.h>

#include <stdlib.h>

int main() {

int a, b, c, c1, c2, c3;

char istriangle;

do {

printf("Enter three sides of the triangle a, b & c: ");

scanf("%d%d%d", &a, &b, &c);

c1 = (a >= 1 && a <= 10);

c2 = (b >= 1 && b <= 10);

c3 = (c >= 1 && c <= 10);

if (!c1) {

printf("Value of a out of range\n");

}

if (!c2) {

printf("Value of b out of range\n");

}

if (!c3) {

printf("Value of c out of range\n");

}

} while (!(c1 && c2 && c3));

if ((a < b + c) && (b < a + c) && (c < a + b)) {

istriangle = 'y';

} else {

istriangle = 'n';

}

if (istriangle == 'y') {

if ((a == b) && (b == c)) {

printf("Equilateral triangle\n");

} else if ((a != b) && (b != c) && (c != a)) {

printf("Scalene triangle\n");

} else {

printf("Isosceles triangle\n");

}

} else {

printf("Not a triangle\n");

}

return 0;

}

Test Report:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Case Id** | **Description** | **Input Data** | |  | **Expected Output** | **Actual Output** | **Comments** |
| **a** | **b** | **c** |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

5. Design, develop, code and run the program in any suitable language to solve the commission problem. Analyze it from the perspective of dataflow testing, derive different test cases, execute these test cases and discuss the test results.

ALGORITHM

STEP 1: Define lockPrice=45.0, stockPrice=30.0, barrelPrice=25.0

STEP2: Input locks

STEP3: while(locks!=-1) ‘input device uses -1 to indicate end of data goto

STEP 12

STEP4:input (stocks, barrels)

STEP5: compute lockSales, stockSales, barrelSales and sales

STEP6: output(“Total sales:” sales)

STEP7: if (sales > 1800.0) goto STEP 8 else goto STEP 9

STEP8: commission=0.10\*1000.0; commission=commission+0.15 \* 800.0;

commission = commission + 0.20 \* (sales-1800.0)

STEP9: if (sales > 1000.0) goto STEP 10 else goto STEP 11

STEP10: commission=0.10\* 1000.0; commission=commission + 0.15 \*

(sales-1000.0)

STEP11: Output(“Commission is $”, commission)

STEP12: exit

PROGRAM CODE:

#include <stdio.h>

#include <conio.h>

int main() {

int locks, stocks, barrels, tlocks, tstocks, tbarrels;

float lprice, sprice, bprice, sales, comm;

int c1, c2, c3, temp;

lprice = 45.0;

sprice = 30.0;

bprice = 25.0;

tlocks = 0;

tbarrels = 0;

tstocks = 0;

printf("Enter the number of locks and to exit the loop enter -1 for locks\n");

scanf("%d", &locks);

while (locks != -1) {

c1 = (locks <= 0) || (locks > 70);

printf("Enter the number of stocks and barrels: ");

scanf("%d%d", &stocks, &barrels);

c2 = (stocks <= 0 || stocks > 80);

c3 = (barrels <= 0 || barrels > 90);

if (c1) {

printf("Value of locks not in the range 1----70\n");

} else {

temp = tlocks + locks;

if (temp > 70) {

printf("New total locks=%d not in the range 1---70\n", temp);

} else {

tlocks = temp;

}

}

printf("Total locks=%d\n", tlocks);

if (c2) {

printf("Value of stocks not in the range 1----80\n");

} else {

temp = tstocks + stocks;

if (temp > 80) {

printf("New total stocks=%d not in the range 1---80\n", temp);

} else {

tstocks = temp;

}

}

printf("Total stocks=%d\n", tstocks);

if (c3) {

printf("Value of barrels not in the range 1----90\n");

} else {

temp = tbarrels + barrels;

if (temp > 90) {

printf("New total barrels=%d not in the range 1---90\n", temp);

} else {

tbarrels = temp;

}

}

printf("Total barrels=%d\n", tbarrels);

printf("Enter the number of locks & to exit the loop enter -1 for locks\n");

scanf("%d", &locks);

}

printf("Total locks=%d \nTotal stocks=%d \nTotal barrels=%d\n", tlocks, tstocks, tbarrels);

sales = lprice \* tlocks + sprice \* tstocks + bprice \* tbarrels;

printf("\nTotal sales=%f\n", sales);

if (sales > 0) {

if (sales > 1800.0) {

comm = 0.10 \* 1000;

comm = comm + 0.15 \* 800;

comm = comm + (0.20 \* (sales - 1800));

} else if (sales > 1000) {

comm = 0.10 \* 1000;

comm = comm + 0.15 \* (sales - 1000);

} else {

comm = 0.10 \* sales; // Correct calculation for sales <= 1000

}

printf("The total commission is %f\n", comm);

} else {

printf("There is no sales\n");

}

return 0;

}

Test Report:

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **Case Id** | **Description** | **Input Data** | |  | **Expected Output** | **Actual Output** | **Comments** |
| **a** | **b** | **c** |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |  |

6 .Design, develop, code and run the program in any suitable language to implement the binary search algorithm. Determine the basis paths and using them derive different test cases, execute these test cases and discuss the test results

ALGORITHM

Step 1: Input value of ‘n’. Enter ‘n’ integer numbers in array int mid;

Step 2: Initialize low = 0, high = n -1

Step 3: until ( low <= high ) do

mid = (low + high) / 2

if ( a[mid] == key )

then do Step 5

else if ( a[mid] > key )

then do

high = mid - 1

else

low = mid + 1

Step 4: Print unsuccessful search do step 6.

Step 5: Print Successful search. Element found at position mid+1.

Step 6: Stop.

PROGRAM CODE:

#include <stdio.h>

int bin(int x[], int low, int high, int key) {

int mid;

while (low <= high) {

mid = (low + high) / 2;

if (x[mid] == key)

return mid; // Return the index of the found element

if (x[mid] < key)

low = mid + 1;

else

high = mid - 1;

}

return -1; // Element not found

}

int main() {

int a[20], n, key, i, success;

printf("Enter the value of n:\n");

scanf("%d", &n);

if (n > 0) {

printf("Enter %d elements in ASCENDING order\n", n);

for (i = 0; i < n; i++) {

scanf("%d", &a[i]);

}

printf("Enter the key element to be searched\n");

scanf("%d", &key);

success = bin(a, 0, n - 1, key);

if (success >= 0)

printf("Element found at index %d \n", success);

else

printf("Element not found\n");

} else {

printf("Number of elements should be greater than zero \n");

}

return 0;

}

Test Report: