**1) Get three values x, y, z and write a program to print 1 if x is the middle value, 2 if y is the middle value and 3 if z is the middle value. Assume that all three variables (x, y, z) are distinct and have different values.**

#include <stdio.h>

int main() {

int x, y, z;

printf("Enter three distinct values: ");

scanf("%d %d %d", &x, &y, &z);

// If y is the middle value

if ((x < y && y < z) || (z < y && y < x)) {

printf("\n 2");

// If x is the middle value

} else if ((y < x && x < z) || (z < x && x < y)) {

printf("\n 1");

// If z is the middle value

} else {

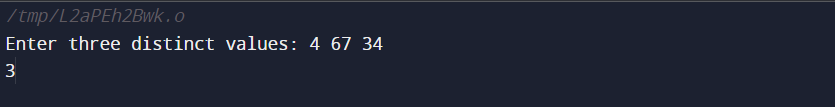
printf("\n 3");

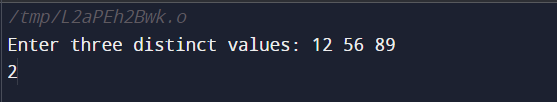
}

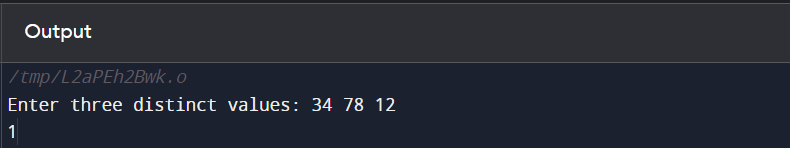
return 0;

}

OUTPUT:







2) **A password is said to be strong if it satisfies the following criteria:**

**It contains at least one lowercase English character.**

**It contains at least one uppercase English character.**

**It contains at least one special character.**

**The special characters are: !@#$%^&\*()-+**

**Its length is at least 8.**

**It contains at least one digit. Given a string, find its strength.**

#include <stdio.h>

#include <stdbool.h>

/\*A password is said to be strong if it satisfies the following criteria:

It contains at least one lowercase English character.

It contains at least one uppercase English character.

It contains at least one special character.

The special characters are: !@#$%^&\*()-+

Its length is at least 8.

It contains at least one digit. Given a string, find its strength.

\*/

bool hasLowerCase(char password[]) {

for (int i = 0; password[i] != '\0'; i++) {

if (password[i] >= 'a' && password[i] <= 'z') {

return true;

}

}

}

bool hasUpperCase(char password[]) {

for (int i = 0; password[i] != '\0'; i++) {

if (password[i] >= 'A' && password[i] <= 'Z') {

return true;

}

}

}

bool hasDigit(char password[]) {

for (int i = 0; password[i] != '\0'; i++) {

if (password[i] >= '0' && password[i] <= '9') {

return true;

}

}

}

bool hasSpecialChar(char password[]) {

char specialChars[] = "!@#$%^&\*()-+";

int length = strlen(specialChars);

for (int i = 0; password[i] != '\0'; i++) {

for (int j = 0; j < length; j++) {

if (password[i] == specialChars[j]) {

return true;

}

}

}

return false;

}

int main() {

char passwrd[20];

printf("Enter your password: ");

scanf("%s", passwrd);

if(strlen(passwrd)>=8)

{

if(hasLowerCase(passwrd))

{

if(hasUpperCase(passwrd))

{

if(hasDigit(passwrd))

{

if (hasSpecialChar(passwrd))

{

printf("Password is strong");

}

else

{

printf("Password must contain atleast 1 special character");

printf("\nPassword is weak");

}

}

else

{

printf("Password must contain atleast 1 digit");

printf("\nPassword is weak");

}

}

else

{

printf("Password must contain atleast 1 uppercase alphabet");

printf("\nPassword is weak");

}

}

else

{

printf("Password must contain atleast 1 lowercase alphabet");

printf("\nPassword is weak");

}

}

else

{

printf("Invalid password length");

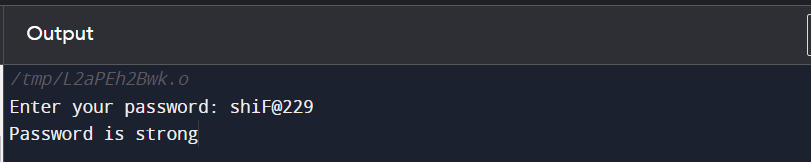
printf("\nPassword is weak");

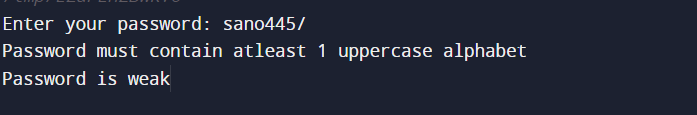
}

return 0;

}

OUTPUT:





3. **A firm creates projects for which a certain number of hours are needed. The firm has a certain number of days. During 10% of the days, the workers are being trained and cannot work on the project. A normal working day is 8 hours long. The project is important for the firm and every worker must work on it with overtime of 2 hours per day. The hours must be rounded down to the nearest integer (for example, 6.98 hours are rounded to 6 hours). Write a program that calculates whether the firm can finish the project on time and how many hours more are needed or left.**

**Input:**

**Accept three integers as input(total number of hours needed,number of days,number of workers).**

**Output:**

**If the time is enough,print "Yes!{the hours left} hours left.".**

**If the time is NOT enough, print "Not enough time!{additional hours} hours needed.**

#include <stdio.h>

int main() {

int totalHoursNeeded, numberOfDays, numberOfWorkers;

printf("Enter total hours needed,number of days,and number of workers:\n");

scanf("%d",&totalHoursNeeded);

scanf("\n%d",&numberOfDays);

scanf("\n%d",&numberOfWorkers);

// 10% training days deduction

int workingDays = numberOfDays - (numberOfDays \* 0.1);

// Normal working hours

int normalWorkHours = workingDays \* numberOfWorkers \* 8;

// Overtime hours

int overtimeHours = workingDays \* numberOfWorkers \* 2;

int totalHoursAvailable = normalWorkHours + overtimeHours;

if (totalHoursAvailable >= totalHoursNeeded) {

int hoursLeft = totalHoursAvailable - totalHoursNeeded;

printf("Yes! %d hours left.\n", hoursLeft);

} else {

int additionalHoursNeeded = totalHoursNeeded - totalHoursAvailable;

printf("Not enough time! %d hours needed.\n", additionalHoursNeeded);

}

return 0;

}

