સી ભાષામાં ડેટા પ્રકાર, પ્રક્રિયકો અને પદાવલિઓ

આ પહેલાના પ્રકરણમાં આપણે કેટલાક સરળ સી પ્રોગ્રામ જોયા અને સી ભાષાના મૂળાક્ષરો તથા ટોકન વિશે પણ અભ્યાસ કર્યો. તેમાનાં એક ટોકન 'આઇડેન્ટિફાયર'(Identifier)ની પણ આપણે ચર્ચા કરેલી. તમામ સી પ્રોગ્રામમાં આઇડેન્ટિફાયરનો ઉપયોગ કરવામાં આવે છે. ધારો કે, પ્રોગ્રામમાં આપણે date નામનો આઇડેન્ટિફાયર વ્યાખ્યાયિત કરવો છે, જે તારીખની કિંમતનો સંગ્રહ કરવા સક્ષમ હોય. આ આઇડેન્ટિફાયરમાં કઈ કિંમતનો સંગ્રહ થવો જોઈએ ? આપણે તેમાં 12.50 કિંમત ઉમેરી શકીએ ? તો જવાબ છે, 'ના'. આ માટે યોગ્ય કિંમત 12 કે 13 જેવી 1થી 31 વચ્ચેની કોઈ પણ પૂર્ણાંક સંખ્યા હશે. આ ઉદાહરણ દર્શાવે છે કે, માત્ર આઇડેન્ટિફાયર નહીં, તેમાં સંગ્રહવામાં આવેલી કિંમત પણ મહત્ત્વની છે. સી ભાષા કેટલાક કી-વર્ડનો ઉપયોગ કરી આઇડેન્ટિફાયરમાં સંગૃહીત કિંમતનો પ્રકાર નિશ્ચિત કરે છે. આ કી-વર્ડ નિશ્ચિત પ્રકારના ડેટા માટે રચવામાં આવેલા હોવાથી તેને ડેટા પ્રકાર (Data type) કહે છે. સી ભાષામાં ઉપલબ્ધ જુદાં જુદાં ડેટા પ્રકારો વિશે આપણે આ પ્રકરણમાં ચર્ચા કરીશું.

ડેટા પ્રકાર શું છે ? (What is Data Type ?)

આઇડેન્ટિફાયરમાં સંગ્રહ કરી શકાય તે કિંમતના પ્રકારને ડેટા પ્રકાર (Data type) તરીકે ઓળખવામાં આવે છે. જો આઇડેન્ટિફાયર dateમાં 12 સંખ્યાનો સંગ્રહ કરવામાં આવે તો તેનો ડેટા પ્રકાર પૂર્ણાંક છે તેમ કહેવાય. આ જ રીતે આઇડેન્ટિફાયર amountને 99.50 કિંમત આપેલ હોય તો તેનો ડેટા પ્રકાર અપૂર્ણાંક છે તેમ કહેવાય.

સી ભાષા વિગતને તેની કિંમત સાથે જોડવા માટે કી-વર્ડનો ઉપયોગ કરે છે. આ કી-વર્ડ બે વસ્તુઓ વ્યાખ્યાયિત કરે છે : આઇડેન્ટિફાયરમાં સંગૃહીત કિંમતનો પ્રકાર અને આઇડેન્ટિફાયર દ્વારા જરૂરી એવી મેમરી જગ્યા. સી ભાષામાં દરેક ડેટા પ્રકારને નિશ્ચિત મેમરી સ્થાન આપવામાં આવે છે. તેનો નિર્દેશ બાઇટ(byte)માં કરવામાં આવે છે. 8 બીટ(Bit)ના સમૂહને 1 બાઇટ કહે છે.

સી ભાષાના મૂળભૂત ડેટા પ્રકાર (Basic Data types of C)

સી ભાષા પૂર્ણાંક (integer), અપૂર્ણાંક (decimal) અને અક્ષર(character)ના નામે ઓળખાતા ડેટા પ્રકારને સમર્થન આપે છે. આ ડેટા પ્રકારને અનુક્રમે int, float અને char કી-વર્ડ દ્વારા રજૂ કરવામાં આવે છે. સી ભાષામાં આ ઉપરાંત void નામનો એક અન્ય પ્રાથમિક ડેટા પ્રકાર પણ આપવામાં આવ્યો છે. આઇડેન્ટિફાયરને તેના ડેટા પ્રકાર સાથે સાંકળવા માટે નીચે આપેલ વાક્યરચનાનો ઉપયોગ કરવામાં આવે છે.

Data type identifier 1 [, identifier 2, identifier 3, ..., identifier n];

અહીં, ચોરસ કૌંસમાં આપવામાં આવેલ લખાણ વૈકલ્પિક છે. હવે આપણે તમામ મૂળભૂત ડેટા પ્રકારો વિશે માહિતી મેળવીએ.

પૂર્ણાંક (Integer)

આ પહેલાના મુદ્દામાં આપણે જોયું કે "date" આઇડેન્ટિફાયરને 12 કિંમત આપી શકાય છે. અહીં 12 એ પૂર્ણાંક સંખ્યા છે. ધન કે ઋણ સંપૂર્ણ સંખ્યા કે જેમાં અપૂર્ણાંક ભાગ આપવામા આવ્યો નથી તેને પૂર્ણાંક કહે છે. –99, 12, –10, 900, 30000 વગેરે પૂર્ણાંક સંખ્યાનાં ઉદાહરણો છે. પૂર્ણાંક પ્રકારના ચલની ઘોષણા માટે નીચે જણાવેલ વાક્યરચનાનો ઉપયોગ કરી શકાય છે :

int identifier 1, [identifier 2, identifier 3, ..., identifier n];

આ વિધાનને સી ભાષામાં ઘોષણા વિધાન (declaration statement) કહે છે. ઘોષણા વિધાનનાં કેટલાંક ઉદાહરણ નીચે મુજબ છે :

```
int roll_number;
int date, month, year;
```

પ્રથમ વિધાન "roll_number" નામ સાથે એક આઇડેન્ટિફાયરની ઘોષણા કરે છે, જે પૂર્ણાંક સંખ્યાનો સંગ્રહ કરવા સક્ષમ છે. બીજું વિધાન પૂર્ણાંક સંખ્યાનો સંગ્રહ કરી શકે તેવા ત્રણ આઇડેન્ટિફાયર "date", "month" અને "year"ની ઘોષણા કરે છે.

ANSI સી માં int ડેટા પ્રકાર 4 બાઇટ જેટલા મેમરી સ્થાનનો ઉપયોગ કરે છે. તેનો વિસ્તાર –2147483648થી +21474836411 છે. અત્યાર સુધીમાં વ્યાખ્યાયિત કરેલા તમામ ચલ ચિહ્નિત (નિશાનીવાળા –signed) છે એટલે કે તેમાં ધન અને ઋણ બંને પ્રકારની કિંમતોનો સંગ્રહ કરી શકાય છે. ક્યારેક આપણને ઋણ સંખ્યાઓની બિલકુલ જરૂર ન હોય એમ પણ બને. અહીં આપણે ઘોષિત કરેલા આ એક પણ ચલમાં ઋણ સંખ્યાનો સંગ્રહ કરી શકાય તેમ નથી. સી ભાષામાં ચલ સાથે માત્ર ધન સંખ્યાઓને જ સાંકળી શકાય તે પણ શક્ય છે. આ માટે unsigned કી-વર્ડનો ઉપયોગ કરવામાં આવે છે. ઉપરના ઉદાહરણને હવે નીચે દર્શાવ્યા મુજબ સુધારી શકાય :

```
unsigned int roll_number;

unsigned int date, month, year;
```

આ વિધાનો દર્શાવે છે કે 'roll_number', 'date', 'month' અને 'year' પૂર્ણાંક પ્રકારના આઇડેન્ટિફાયર છે, જે માત્ર ધન સંખ્યાઓનો સંગ્રહ કરવાની ક્ષમતા ધરાવે છે. પૂર્ણાંક ડેટા પ્રકારની સંક્ષિપ્ત માહિતી કોષ્ટક 11.1માં આપવામાં આવી છે.

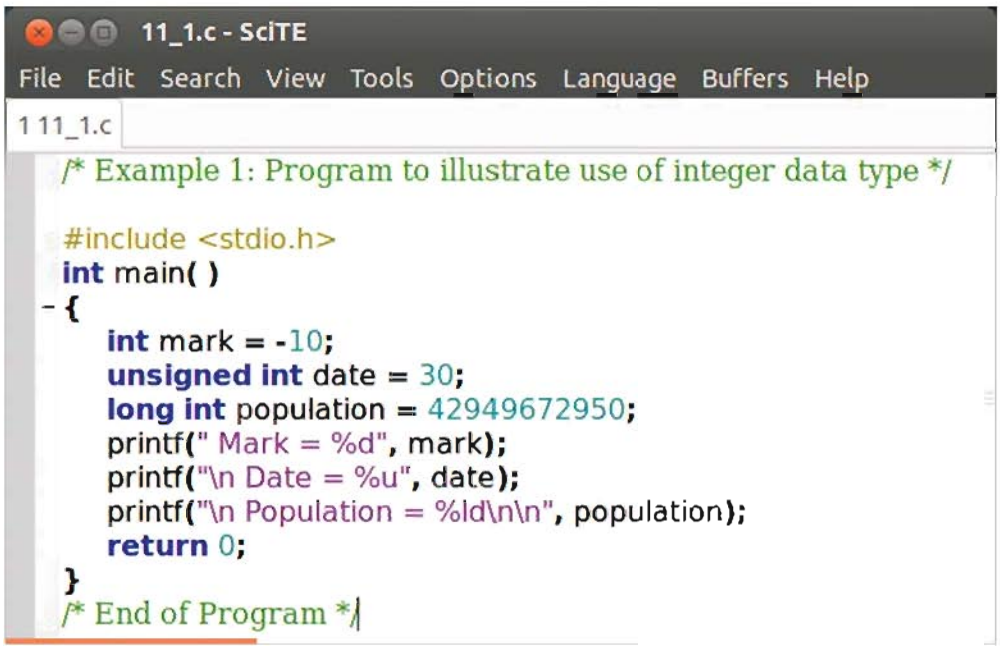
Data Type	Range	Bytes required	Example
int	–2147483648 to +2147483647	4	int balance_amount;
unsigned int	0 to 4294967295	4	unsigned int counter;

કોષ્ટક 11.1 : પૂર્ણાંક ડેટા પ્રકાર

કોષ્ટક 11.1માં આપેલ કિંમતનો વિસ્તાર ઉપયોગમાં લેવામાં આવેલા બાઇટની સંખ્યા પર અવલંબે છે. ચિહ્નિત (signed) પૂર્ણાંક માટે કિંમતોનો વિસ્તાર $-2^{(n-1)}$ to $+2^{(n-1)}-1$ દ્વારા ગણી શકાય છે. અહીં "n" એ જરૂરી બીટની સંખ્યાનો નિર્દેશ કરે છે. આ જ રીતે અચિહ્નિત (unsigned) પૂર્ણાંક માટે વિસ્તારની ગણતરી 0થી $2^{n}-1$ દ્વારા કરી શકાય છે. પૂર્ણાંક ડેટા પ્રકારની આગળ long કી-વર્ડ ઉમેરવાથી પૂર્ણાંક સંખ્યાઓનો વિસ્તાર વધારી શકાય છે. intની આગળ long ઉમેરવાથી ચલ માટે 8 બાઇટ જેટલું મેમરી સ્થાન આરક્ષિત કરવામાં આવે છે. આ પ્રકારના ચલને નીચે જણાવેલ વાક્યરચના દ્વારા વ્યાખ્યાયિત કરી શકાય :

```
long int population;
```

પૂર્ણાંક ડેટા પ્રકારના જુદા જુદા પ્રકારનાં સ્વરૂપો જોયા પછી હવે એક નમૂનારૂપ પ્રોગ્રામનો અભ્યાસ કરીએ. પૂર્ણાંક ડેટા પ્રકારના ઉપયોગ દર્શાવતા પ્રોગ્રામનું કોડ લિસ્ટિંગ આકૃતિ 11.1માં આપેલ છે.



આકૃતિ 11.1 : પૂર્ણાંક ડેટા પ્રકારનો ઉપયોગ દર્શાવતો પ્રોગ્રામ

સમજૂતી (Explanation)

આકૃતિ 11.1માં આવેલા દરેક વિધાનની સમજૂતી મેળવીએ. ઉઘડતા છગડિયા કૌંસ પછીનું પ્રથમ વિધાન "mark" નામના પૂર્ણાંક ચલને ઘોષિત કરે છે. આ જ વિધાનમાં ચલની કિંમત –10 પણ આપી છે. બીજું વિધાન "date" નામના પૂર્ણાંક ચલની ઘોષણા કરી તેમાં કિંમત 30 આપે છે. ત્રીજું વિધાન "population" નામના ચલને ઉચ્ચ વિસ્તાર સાથે ઘોષિત કરી તેમાં 42949672950 કિંમતનો સંગ્રહ કરે છે. પછીના ત્રણ વિધાનો દ્વારા આ ત્રણ ચલની કિંમતોને સ્ક્રીન પર દર્શાવવામાં આવે છે. ઉદાહરણ 11.1નું પરિણામ આકૃતિ 11.2માં દર્શાવ્યું છે.

આકૃતિ 11.2 : ઉદાહરણ 11.1નું પરિણામ

અપૂર્ણાંક (Real)

આપણને કેટલીકવાર પૂર્ણાંકને બદલે અપૂર્ણાંક સંખ્યાઓનો ઉપયોગ કરવાની જરૂર પડે છે. ઉદાહરણ તરીકે, કોઈ વ્યક્તિએ ચુકવવાની રકમ 95 કે 95.50 હોઈ શકે. આ પરિસ્થિતિમાં int ડેટા પ્રકાર કાર્ય કરી શકતો નથી. સી ભાષામાં અપૂર્ણાંક સંખ્યાઓનો ઉપયોગ કરવા માટે float કી-વર્ડ દ્વારા વ્યાખ્યાયિત કરવામાં આવતો ડેટા પ્રકાર આપવામાં આવ્યો છે. તે 4 બાઇટ જેટલા સંગ્રાહક સ્થાનનો ઉપયોગ કરે છે. અપૂર્ણાંક સંખ્યાઓ દશાંશચિહ્ન પછી 6 અને દશાંશચિહ્ન પહેલાં 7 અંકો જેટલી ચોક્સાઈ ધરાવે છે. જો એથી વધુ ચોક્સાઈની જરૂર હોય તો floatના સ્થાને double કી-વર્ડનો ઉપયોગ કરવામાં આવે છે. તે float ડેટા પ્રકારનું વિસ્તરણ છે. double કિંમતો 8 બાઇટનો ઉપયોગ કરે છે અને દશાંશ પછી 16 તથા દશાંશ પહેલાં 17 અંકોની ચોક્સાઈ ધરાવે છે. અપૂર્ણાંક ચલનાં ઉદાહરણ નીચે મુજબ છે :

float amount_to_pay;

double balance_amount;

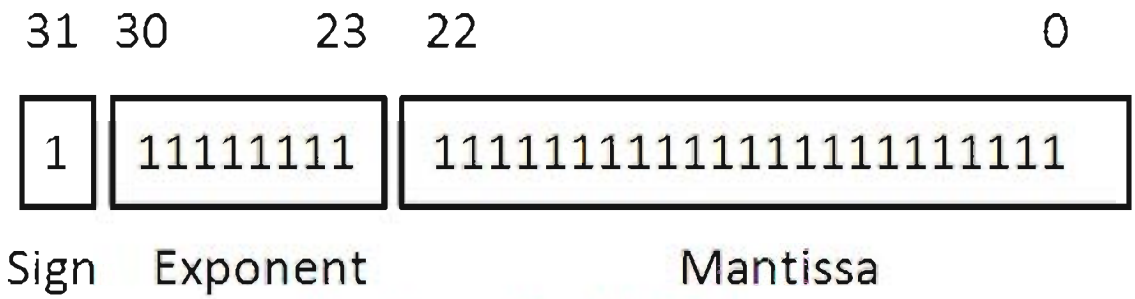
પ્રથમ વિધાન અપૂર્ણાંક સંખ્યાનો સંગ્રહ કરવા માટે સક્ષમ એવા "amount_to_pay" નામના ચલને ઘોષિત કરે છે. બીજું વિધાન "balance_amount" નામના ચલને ઘોષિત કરે છે જે વધુ વિસ્તાર અને ચોકસાઈ સાથે અપૂર્ણાંક સંખ્યાનો સંગ્રહ કરવા સક્ષમ છે.

double કી-વર્ડની આગળ long શબ્દ પણ ઉમેરી શકાય છે. આમ કરવાથી doubleનો વિસ્તાર વધારી શકાય છે. float ડેટા પ્રકારની સંક્ષિપ્ત સમજૂતી કોષ્ટક 11.2માં આપવામાં આવી છે.

Data Type	Range	Significant Digits	Bytes required	Example
float	+/–3.4e–38 to +/–3.4e+38	6	4	float price;
double	+/–1.7e-308 to +/–1.7e+308	16	8	double tot_price;
long double	+/–3.4e–4932 to +/–1.1e+4932	16	16	long double credit;

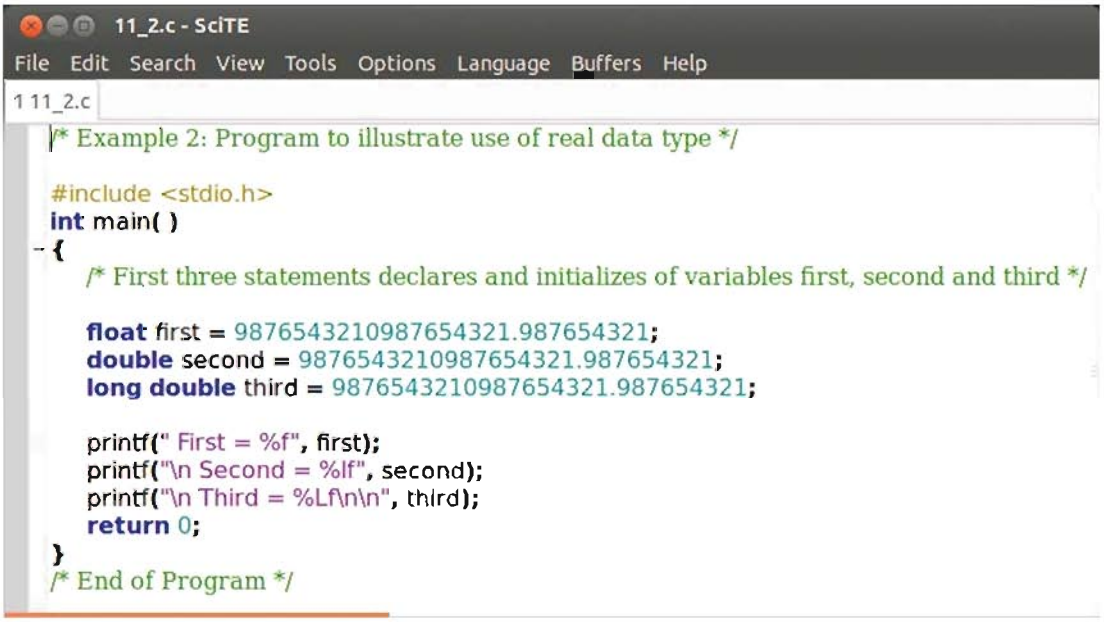
કોષ્ટક 11.2 : float ડેટા પ્રકાર

કોષ્ટક 11.2માં વિસ્તારને વૈજ્ઞાનિક સ્વરૂપમાં દર્શાવ્યો છે. સી ભાષા વૈજ્ઞાનિક સ્વરૂપે રહેલી સંખ્યાને પણ સમર્થન આપે છે. ઉદાહરણ તરીકે 95.50 સંખ્યાને 0.9550e2 સ્વરૂપમાં પણ રજૂ કરી શકાય છે. અહીં 0.9550ને મેન્ટિસા (mantissa) તરીકે તથા 2ને એક્સ્પોનન્ટ (exponent) તરીકે ઓળખવામાં આવે છે. અપૂર્ણ સંખ્યા ધન કે ઋણ હોઈ શકે છે. આકૃતિ 11.4માં અપૂર્ણાંક સંખ્યાની મેમરી રજૂઆત દર્શાવી છે.



આકૃતિ 11.3 : અપૂર્ણાંક સંખ્યાની મેમરી રજૂઆત

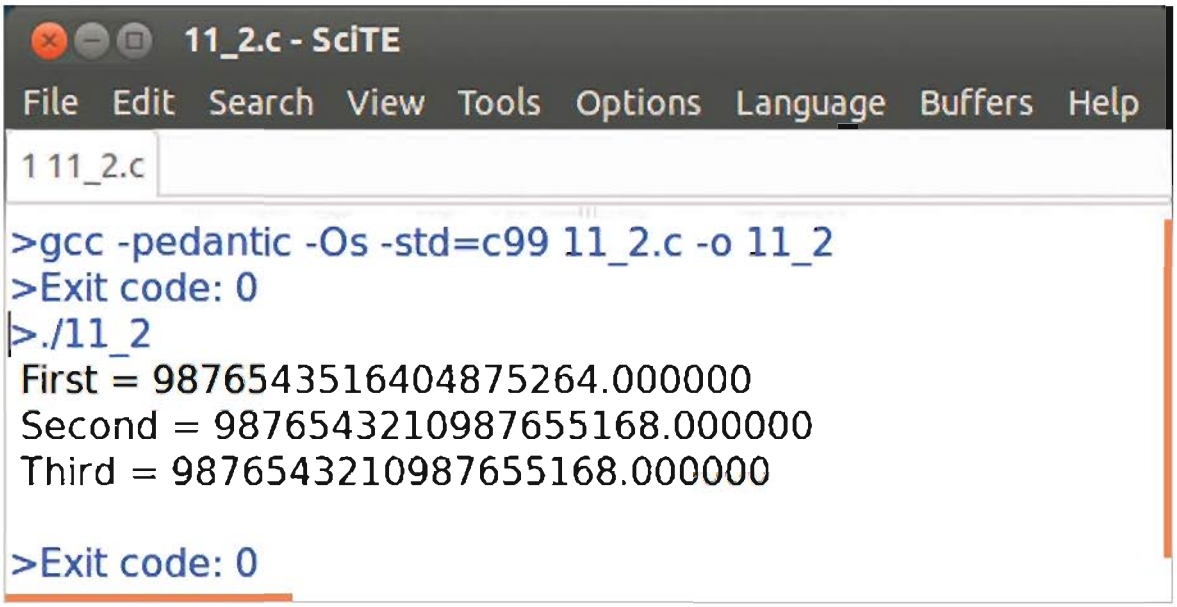
હવે, float, double અને long double ડેટા પ્રકારનો ઉપયોગ દર્શાવતું એક ઉદાહરણ જોઈએ. આકૃતિ 11.4માં float ડેટા પ્રકારને સ્પષ્ટ કરતા પ્રોગ્રામનું કોડ લિસ્ટિંગ દર્શાવ્યું છે.



આકૃતિ 11.4 : float ડેટા પ્રકારને સ્પષ્ટ કરતો પ્રોગ્રામ

સમજૂતી (Explanation)

ઉઘડતા છગડિયા કૌંસ પછીનાં ત્રણ વિધાનો "first", "second" અને "third" નામના ત્રણ ચલને ઘોષિત કરે છે. આ દરેક ચલમાં અપૂર્ણ સંખ્યા 9876543210987654321.987654321નો સંગ્રહ કરવામાં આવ્યો છે. પછીનાં ત્રણ વિધાનો આ ચલની કિંમતોને સ્ક્રીન પર દર્શાવે છે. આકૃતિ 11.5માં ઉદાહરણ 11.2નું પરિણામ દર્શાવ્યું છે.



આકૃતિ 11.5 : ઉદાહરણ 11.2નું પરિણામ

અહીં જોઈ શકાય છે કે, ચલ 'first', 'second' અને 'third'ની કિંમતો અનુક્રમે 9876543516404872526 4.000000, 9876543210987655168.000000 અને 9876543210987655168.000000 દર્શાવવામાં આવી છે. પરિણામને ધ્યાનથી જોતાં જાણી શકાય છે કે "first" ચલની મૂળ કિંમત સાથે પ્રથમ માત્ર 7 અંકોની સમાનતા છે. એ જ પ્રમાણે "second" અને "third" ચલના પ્રથમ માત્ર 15 અંકો જ સમાન છે. આમ થવાનું કારણ એ છે કે float ડેટા પ્રકાર દશાંશ પછી 6 અને દશાંશ પહેલાં 7 અંકોની ચોક્સાઈ ધરાવે છે. આ જ રીતે double અને long float ડેટા પ્રકાર દશાંશ પછી 15 અને દશાંશ પહેલાં 16 અંકોની ચોક્સાઈ ધરાવે છે.

અપૂર્ણાંક સંખ્યાઓનો ઉપયોગ કેટલીકવાર ચોકસાઈને હાનિ પહોંચાડે છે. તેથી આપણે અપેક્ષિત નિશ્ચિત પરિણામ મેળવી શકતા નથી. માટે જ પરિણામની વધુ ચોકસાઈ માટે ઉચ્ચ પરિશુદ્ધતા (precision) ધરાવતા ડેટા પ્રકારનો ઉપયોગ હિતાવહ છે. અપૂર્ણાંક અંકોનો ઉપયોગ કરવાનો લાભ એ છે કે તે પૂર્ણાંક અંકોની સરખામણીમાં કિંમતોનો વ્યાપક વિસ્તાર રજૂ કરે છે.

અક્ષર (character)

ઉપરના બંને કિસ્સાઓમાં આપણે આંકડાકીય વિગતોનો સંગ્રહ કર્યો. પરંતુ ધારો કે આપણે પુરુષ માટે 'M' (male) અને સ્ત્રી માટે 'F' (female) જેવી જાતિવાચક સંજ્ઞા અથવા તો શહેરનાં નામ જેવી વિગતોનો સંગ્રહ કરવાની જરૂર હોય તો ? અહીં આપણી પાસે માત્ર મૂળાક્ષરો છે. આ પ્રકારની કિંમતોનો સંગ્રહ int કે float દ્વારા કરી શકાતો નથી. આ પ્રકારની કિંમતોનો સંગ્રહ કરવા માટે char કી-વર્ડનો ઉપયોગ કરવામાં આવે છે. તે 1 બાઇટ જેટલા મેમરી સ્થાનનો ઉપયોગ કરે છે. દરેક અક્ષર ASCII (American Standard Code for Information Interchange) નામે ઓળખતાં પૂર્ણાંક સાથે સંકળાયેલ હોય છે. ASCII કિંમતોની વિગતો પરિશિષ્ટ-II માં આપવામાં આવી છે. પૂર્વ નિર્ધારિત રીતે char અચિહ્નિત(Unsigned) છે. ચિહ્નિત (signed) char પણ શક્ય છે. કોષ્ટક 11.3માં char ડેટા પ્રકાર વિશે ટૂંકી સમજૂતી આપવામાં આવી છે.

Data Type	Range	Bytes required	Example
char	–128 to + 127	1	char gender;
unsigned char	0 to 255	1	unsigned char choice;

કોષ્ટક 11.3 : char ડેટા પ્રકાર

char ડેટા પ્રકારને સ્પષ્ટ કરતા પ્રોગ્રામનું કોડ લિસ્ટિંગ આકૃતિ 11.6માં દર્શાવ્યું છે.

11_3.c - SciTE

File Edit Search View Tools Options Language Buffers Help

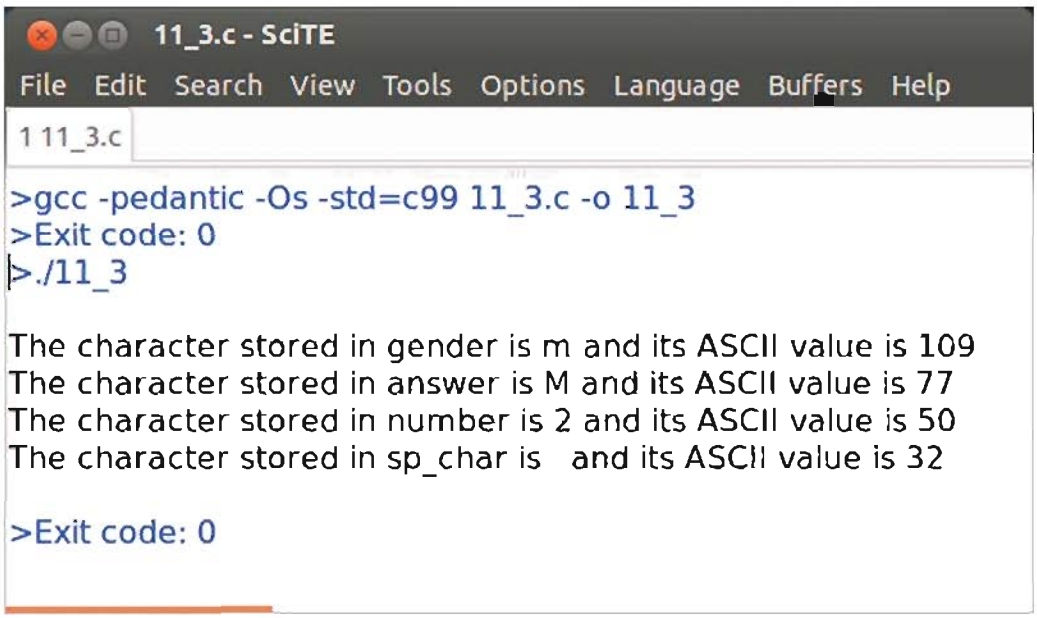
1 11_3.c

```c
/*Example 3: Program to illustrate use of character data type */

#include <stdio.h>
int main( )
{
    char gender = 'm';
    char answer = 'M';
    char number = '2';
    char sp_char = ' ';
    printf("\nThe character stored in gender is %c and its ASCII value is %d", gender, gender);
    printf("\nThe character stored in answer is %c and its ASCII value is %d", answer, answer);
    printf("\nThe character stored in number is %c and its ASCII value is %d", number, number);
    printf("\nThe character stored in sp_char is %c and its ASCII value is %d\n\n", sp_char, sp_char);
    return 0;
}
/* End of Program */
```

આકૃતિ 11.6 : char ડેટા પ્રકાર દર્શાવતો પ્રોગ્રામ

સમજૂતી (Explanation)

ઉઘડતા છગડિયા કૌંસ પછીનાં પ્રથમ ચાર વિધાનો અનુક્રમે "gender", "answer", "number" અને "sp_char" નામના ચાર ચલ ઘોષિત કરી તેને અનુક્રમે 'm', 'M', '2' અને ' ' (ખાલી જગ્યા) આપશે. પછીનાં ચાર વિધાનો આ ચલમાં આવેલી કિંમતોને તેની ASCII કિંમત સાથે સ્ક્રીન પર દર્શાવશે. ઉદાહરણ 11.3નું પરિણામ આકૃતિ 11.7માં દર્શાવ્યું છે.



આકૃતિ 11.7 : ઉદાહરણ 11.3નું પરિણામ

સ્મૉલ m અને કૅપિટલ Mની ASCII કિંમતો વચ્ચેના તફાવતની નોંધ કરો. આપણે જોઈ શકીએ છીએ કે અંકોનો ઉપયોગ અક્ષરોના સ્વરૂપે પણ કરી શકાય છે. સંખ્યા 2નો અક્ષર તરીકે ઉપયોગ કરવા માટે તેને '2' સ્વરૂપે લખવામાં આવે છે. અહીં એ પણ જુઓ કે sp_char ચલમાં આવેલ કિંમત દૃશ્યમાન ન હોવા છતાં તેનો આસ્કી અંક દૃશ્યમાન છે.

ખાલી વિગતોનો ગણ (Empty data set)

સી ભાષામાં void કી-વર્ડ દ્વારા એક વિશિષ્ટ ડેટા પ્રકાર પૂરો પાડવામાં આવે છે. આ ડેટા પ્રકાર કોઈ કિંમતનો સંગ્રહ કરતો નથી માટે તેનો 'ખાલી ગણ' તરીકે નિર્દેશ કરવામાં આવે છે. તેનો ઉપયોગ મુખ્યત્વે વિધેયની પરત કિંમત દર્શાવવા માટે કરવામાં આવે છે. આગળના પ્રકરણમાં જણાવ્યું તે મુજબ સી પ્રોગ્રામ વિધેયોનો સમૂહ છે. સી પ્રોગ્રામના વિધેય કોઈ કિંમત પરત કરી શકે છે. જો આપણે ઇચ્છતા હોઈએ કે વિધેય કોઈ કિંમત પરત ન કરે તો તેના નામની પહેલાં void કી-વર્ડ ઉમેરવામાં આવે છે.

આગળનાં તમામ ઉદાહરણોમાં પ્રોગ્રામનું અંતિમ વિધાન return 0; છે. આ વિધાન કંપાઇલરને પ્રોગ્રામમાંથી બહાર નીકળવાનો નિર્દેશ કરે છે. જો આ વિધાન ન લખવામાં આવે તો ચેતવણીનો સંદેશ "Function should return a value" રજૂ કરવામાં આવે છે. આ સંદેશ ન આવે તે માટે main( )ની આગળ void ઉમેરી શકાય છે.

ચલને કિંમત આપવી (Assigning values to variable)

પ્રોગ્રામના અમલીકરણ દરમિયાન એક વાર ઘોષિત કર્યા બાદ ચલમાં કિંમતો આપી શકાય છે. આ કિંમતો ચલના ડેટા પ્રકારને અનુરૂપ હોવી જોઈએ. નીચેની વાક્યરચના દ્વારા ચલમાં કિંમત આપી શકાય છે :

```
Variable = Value;
```

એક જ વિધાનમાં ચલની ઘોષણા કરી તેમાં કિંમત આપવી પણ શક્ય છે. આમ કરવા માટે નીચેની વાક્યરચનાનો ઉપયોગ કરી શકાય :

```
data type variable = value;
```

અત્યાર સુધી આ પ્રકરણમાં ચર્ચવામાં આવેલાં તમામ ઉદાહરણોમાં ચલનો પ્રારંભ આ તકનિકથી કરવામાં આવ્યો છે.

ઉપયોગકર્તા દ્વારા નિર્મિત ડેટા પ્રકાર (User defined data type)

સી ભાષા પરિવર્તનક્ષમ (flexible) છે. તે ઉપયોગકર્તાને હયાત ઘટકોમાંથી નવા ઘટકો બનાવવાની સુવિધા આપે છે તથા મૂળભૂત ડેટા પ્રકારનો ઉપયોગ કરી નવા ડેટા પ્રકારની રચના કરી શકાય છે. સી ભાષામાં ઉપયોગકર્તા દ્વારા વ્યાખ્યાયિત ડેટા પ્રકારની રચના કરવા માટે typedef અને enum નામના બે કી-વર્ડનો ઉપયોગ કરવામાં આવે છે. બંને પ્રકારના કી-વર્ડ દ્વારા ડેટા પ્રકાર તરીકે ઉપયોગમાં લઈ શકાય તેવા ચલ વ્યાખ્યાયિત કરી શકાય છે.

ટાઈપ ડેફિનેશન (Type definition)

ટાઈપ ડેફિનેશન દ્વારા ડેટા પ્રકાર વ્યાખ્યાયિત કરવા માટેની વાક્યરચના નીચે પ્રમાણે છે :

typedef datatype variable;

અહીં, datatype એ int કે float જેવા ઉપલબ્ધ ડેટા પ્રકારનો નિર્દેશ કરે છે. ઉપલબ્ધ ડેટા પ્રકારને આપવામાં આવનાર નવા નામનો નિર્દેશ variable દ્વારા કરવામાં આવે છે. ખરી રીતે જોતાં, સી ભાષા નવો ડેટા પ્રકાર રચવાની અનુમતિ આપતું નથી. તેને બદલે તે ઉપલબ્ધ ડેટા પ્રકારને ઉપનામ (alias) આપવાની સુવિધા ધરાવે છે. આપણે સામાન્ય રીતે જીવનમાં પણ ઉપનામોનો ઉપયોગ કરીએ છીએ. વ્યક્તિને આપવામાં આવતું હુલામણું નામ તેનું ઉદાહરણ છે. typedefના કેટલાંક ઉદાહરણ નીચે આપેલાં છે :

typedef char alpha;

typedef double twice;

પ્રથમ વિધાન char ડેટા પ્રકારને "alpha" ઉપનામ અપાશે જ્યારે બીજા વિધાન દ્વારા double ડેટા પ્રકારને "twice" ઉપનામ આપવામાં આવશે. આ ઉપનામ આપ્યા બાદ, નીચેનાં વિધાનો દ્વારા અક્ષર માટેનો "choice" નામનો ચલ તથા "amount" નામનો ''ડબલ'' પ્રકારનો ચલ ઘોષિત કરી શકાય.

alpha choice;

twice amount;

ઈન્યુમરેટેડ ડેટા પ્રકાર (Enumerated data type)

સી ભાષામાં enum કી-વર્ડ દ્વારા ઈન્યુમરેટેડ ડેટા પ્રકારની રચના કરી શકાય છે. enumનો ઉપયોગ કરવા માટેની વાક્યરચના નીચે મુજબ છે :

enum identifier {value 1, value 2, value 3, ..., value n};

અહીં enum કી-વર્ડ છે. Identifier એ નવો ડેટા ટાઈપ વ્યાખ્યાયિત કરવા માટે ઉપયોગમાં લેવામાં આવનાર નામ છે. Value 1થી Value n એ આંકડાકીય અચલો 0, 1, 2, ... વગેરેને આપવામાં આવેલ નામ છે. enum દ્વારા વ્યાખ્યાયિત કરવામાં આવેલા ચલમાં છગડિયા કૌંસમાં આપવામાં આવેલી કોઈ પણ એક કે વધુ કિંમત ઉમેરી શકાય છે. enum ના ઉપયોગનું ઉદાહરણ નીચે મુજબ છે :

enum money {rupee, dollar, pound, yen};

enum money currency;

currency = dollar;

પ્રથમ વિધાન ઈન્યુમરેટેડ ડેટા પ્રકાર "money" વ્યાખ્યાયિત કરે છે. બીજા વિધાન દ્વારા money પ્રકારનો ઉપયોગ કરી "currency" નામનો ચલ ઘોષિત કરવામાં આવ્યો છે. છેલ્લા વિધાનમાં currency ચલને "dollar" કિંમત આપવામાં આવી છે જે ''1''ને સમકક્ષ છે. અહીં, rupee, dollar, pound અને yen અનુક્રમે 0, 1, 2 અને 3 આંકડાકીય અચળો રજૂ કરે છે. આ આંકડાકીય અચળો કંપાઇલર દ્વારા આપોઆપ ઉમેરવામાં આવે છે. નવી કિંમતો ઉમેરી આ અચળોને બદલી પણ શકાય છે. ઉદાહરણ તરીકે જો આપણે rupeeને 10, dollarને 50, poundને 75 અને yenને 100 કિંમત આપવી હોય તો પ્રથમ વિધાનને enum money {rupee = 10, dollar = 50, pound = 75, yen = 100}; સ્વરૂપે લખી શકાય. હવે currency = dollar વિધાન currency = 50ને સમકક્ષ બનશે.

ઉપયોગકર્તા દ્વારા નિર્મિત ડેટા પ્રકારોના ઉપયોગથી સી પ્રોગ્રામની વાચનક્ષમતામાં વૃદ્ધિ કરી શકાય છે. તે ડેટા પ્રકારને અર્થપૂર્ણ નામ સાથે વ્યાખ્યાયિત કરવાની સુવિધા પણ આપે છે.

તારવેલા ડેટા પ્રકાર (Derived data type)

આપણે અગાઉ જોઈ ગયા તેમ સી વિસ્તારી શકાય તેવી (Extensible) ભાષા છે. આપણી ઘણી જરૂરિયાતો માટે

મૂળભૂત ડેટા પ્રકારો પૂરતા બની રહે તેમ બને, પરંતુ જટિલતાને કારણે દરેક સમયે તેનો ઉપયોગ કરવો યોગ્ય નથી. ઉદાહરણ તરીકે, ધારો કે આપણે 15 વિદ્યાર્થીઓના વર્ગ(grades)નો સંગ્રહ કરવા ઇચ્છીએ છીએ. આ માટે આપણે 15 ચલ ઘોષિત કરવાની જરૂર પડે. 15 ચલની સારસંભાળ પ્રોગ્રામની જટિલતામાં વધારો કરે. આ જ રીતે જો આપણે 15 વિદ્યાર્થીઓ માટે તેમના અનુક્રમ, જાતિ, નામ અને સરનામા જેવી સંપૂર્ણ વિગતનો સંગ્રહ કરવો હોય તો વધારાના ચલની જરૂર પડે. અહીં ચલના જૂથ કદાચ મુશ્કેલી ઊભી કરે. તેના નિવારણ માટે સી ભાષા ડેટા પ્રકાર તારવવાની સુવિધા પૂરી પાડે છે. એરે (array), સ્ટ્રક્ચર (structure), યુનિયન (union) અને પૉઇન્ટર (pointer) તારવેલાં ડેટા પ્રકારનાં ઉદાહરણ છે.

એરે (Array)

સી ભાષામાં એરે નામના ડેટા બંધારણની રચના શક્ય છે. આ ડેટા બંધારણમાં એકસમાન લાક્ષણિકતા ધરાવતા ચલના સમૂહનો સમાવેશ કરી શકાય છે. એરે વ્યાખ્યાયિત કરવા માટેની વાક્યરચના નીચે મુજબ છે :

data type variable[size];

અહીં datatype એ ડેટાનો કોઈ પણ મૂળભૂત પ્રકાર હોઈ શકે છે. Variable એ એરેનું નામ છે અને size એ એરેમાં આવેલા ઘટકોની કુલ સંખ્યા છે. ઉદાહરણ તરીકે, **char name_of_subject[10];** સી વિધાન name_of_subject નામનો એક 10 કદ ધરાવતો એરે વ્યાખ્યાયિત કરશે. આનો અર્થ એ થાય કે, "name_of_subject" નામનો ચલ વ્યાખ્યાયિત કરી તેમાં 10 અક્ષરોનો સમૂહ સંગ્રહ કરી શકાશે. અહીં એરેનો દરેક ઘટક અક્ષર પ્રકારનો રહેશે.

સી ભાષામાં **char name_of_subject[ ] = "C Language";** પ્રકારનું વિધાન પણ હોઈ શકે છે. અહીં એરેની લંબાઈ કંપાઇલર દ્વારા આપોઆપ નિશ્ચિત કરી લેવામાં આવશે. આપેલ કિસ્સામાં એરેની લંબાઈ 11 હશે. પ્રકરણ 15માં એરે વિશે વિસ્તૃત ચર્ચા કરવામાં આવી છે.

સી ભાષામાં સ્ટ્રક્ચર, યુનિયન અને પૉઇન્ટર જેવા અન્ય તારવેલા ડેટા પ્રકાર પણ ઉપલબ્ધ છે. આ તમામ ડેટા પ્રકારોની ચર્ચા આ પુસ્તકની મર્યાદા બહાર છે.

પ્રક્રિયક અને પદાવલિ (Operators and Expression)

અત્યાર સુધી આપણે સી ભાષામાં ઉપલબ્ધ ડેટા પ્રકારો વિશે જાણકારી મેળવી. હવે આપણે સી ભાષામાં ઉપલબ્ધ જુદા જુદા પ્રક્રિયકો વિશે અભ્યાસ કરીએ. પ્રક્રિયકો અને ઑપરેન્ડ (operand)ના ઉપયોગ દ્વારા પદાવલિ કેવી રીતે બનાવી શકાય તેનો પણ આપણે અભ્યાસ કરીશું. અંતમાં આપણે જોઈશું કે પદાવલિઓનું મૂલ્યાંકન કેવી રીતે કરવામાં આવે છે.

પ્રક્રિયકો (Operators)

શાળામાં પ્રાથમિક ગણિતનો અભ્યાસ કરતી વખતે આપણે બે સંખ્યાઓના સરવાળા કે બાદબાકી જેવી પ્રક્રિયાઓ કરેલી, જેમકે, 5 + 3 અને 9 - 7. અહીં 5, 3, 9 અને 7 અચળો છે જ્યારે '+' અને '–' નિશાનીઓ આ અચળો પરની પ્રક્રિયા વ્યાખ્યાયિત કરે છે.

ઑપરેન્ડ પર કરવામાં આવનાર પ્રક્રિયાને વ્યાખ્યાયિત કરતી નિશાનીને પ્રક્રિયક (operators) તરીકે ઓળખવામાં આવે છે. સી ભાષામાં પ્રક્રિયકોનું વર્ગીકરણ નીચે આપેલ આઠ વર્ગોમાં કરી શકાય :

1. ગાણિતિક પ્રક્રિયકો (Arithmetic Operators)
2. નિરૂપક પ્રક્રિયકો (Assignment Operators)
3. સંબંધસૂચક પ્રક્રિયકો (Relational Operators)
4. વધારા / ઘટાડા સૂચક પ્રક્રિયકો (Increment and Decrement Operators)
5. શરતી પ્રક્રિયકો (Conditional Operator)
6. તાર્કિક પ્રક્રિયકો (Logical Operators)
7. બિટવાઇઝ પ્રક્રિયકો (Bitwise Operators)
8. વિશિષ્ટ પ્રક્રિયકો (Special Operators)

આ તમામ પ્રક્રિયકો વિશે ચર્ચા કરીએ.

ગાણિતિક પ્રક્રિયકો (Arithmetic Operators)

આંકડાકીય ગણતરીઓ કરવા માટે સી ભાષામાં '+', '–', '*', '/' અને '%' પ્રક્રિયકો આપવામાં આવ્યા છે. ઘાતાંક માટેનો કોઈ પ્રક્રિયક સી ભાષામાં ઉપલબ્ધ નથી. કોષ્ટક 11.4માં આ પ્રક્રિયકો તેમના ઉપયોગ સાથે દર્શાવ્યા છે.

પ્રક્રિયક	ઉપયોગ
+	બે સંખ્યાઓનો સરવાળો અથવા યુનરી ધન
-	બે સંખ્યાઓની બાદબાકી અથવા યુનરી ઋણ
*	બે સંખ્યાઓનો ગુણાકાર
/	બે સંખ્યાઓનો ભાગાકાર કરી ભાગફળ રૂપે પરિણામ
%	બે સંખ્યાઓનો ભાગાકાર કરી શેષ રૂપે પરિણામ

કોષ્ટક 11.4 : આંકડાકીય પ્રક્રિયકો

કોષ્ટક 11.4માં આપવામાં આવેલ પ્રથમ ચાર પ્રક્રિયકો વિશે આપણે માહિતગાર છીએ. સી ભાષા મૉડ્યૂલો (Modulo) નામે ઓળખાતા એક અતિરિક્ત પ્રક્રિયક '%'ની સુવિધા પૂરી પાડે છે. બે પૂર્ણાંક સંખ્યાઓના ભાગાકારથી મળતી શેષ ગણવા માટે આ પ્રક્રિયકનો ઉપયોગ કરવામાં આવે છે. અહીં એ નોંધ લેવી જરૂરી છે કે આ પ્રક્રિયકનો ઉપયોગ અપૂર્ણ (float) સંખ્યાઓ સાથે શક્ય નથી.

હવે આપણે આ પ્રક્રિયકોનો ઉપયોગ કેવી રીતે કરવો તે જોઈએ. ઉદાહરણ સ્વરૂપે આપેલ નીચેનું સી વિધાન જુઓ :

```
total_cost = quantity * cost_item;
```

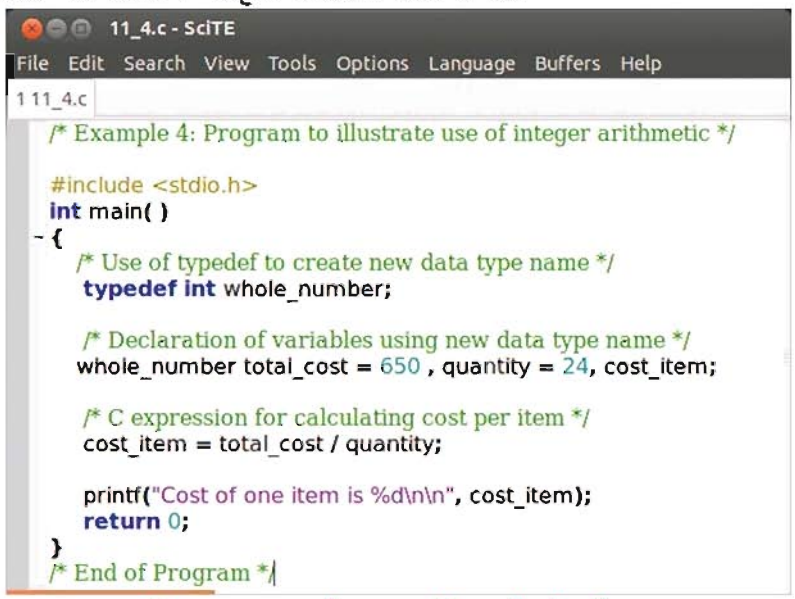
આ સી વિધાન સી પદાવલિ(C expression)ની રચના કરે છે. અહીં 'total_cost', 'quantity' અને 'cost_item'ને સી ચલ તરીકે ઓળખવામાં આવે છે. અહીં જોઈ શકાય છે કે પદાવલિમાં બે પ્રક્રિયકો અને ત્રણ ઑપરેન્ડ આવેલાં છે. આ પદાવલિનું મૂલ્યાંકન કરવામાં આવે ત્યારે બે પ્રક્રિયાઓ થાય છે. પ્રથમ, quantity અને cost_item ચલમાં રહેલી કિંમતોનો ગુણાકાર કરવામાં આવે છે અને બીજું, ગુણાકારથી મળેલા પરિણામનો total_cost નામના ચલમાં સંગ્રહ કરવામાં આવે છે.

પદાવલિમાં ઉપયોગમાં લેવામાં આવેલા ઑપરેન્ડને આધારે ગણતરીને ત્રણ વર્ગોમાં વહેંચી શકાય : પૂર્ણાંક ગણતરી (integer arithmetic), અપૂર્ણાંક ગણતરી (real arithmetic) અને મિશ્ર પ્રકારની ગણતરી (mixed mode arithmetic).

પૂર્ણાંક ગણતરી (Integer Arithmetic)

જો પદાવલિમાં ઉપયોગમાં લેવામાં આવેલ ઑપરેન્ડ ધન કે ઋણ પૂર્ણ સંખ્યાઓ હોય તો તેને પૂર્ણાંક ગણતરી (integer arithmetic) તરીકે ઓળખવામાં આવે છે. અહીં, પદાવલિને પૂર્ણાંક પદાવલિ (integer expression) કહે છે. પૂર્ણાંક પદાવલિનું પરિણામ હંમેશાં પૂર્ણાંક મળે છે.

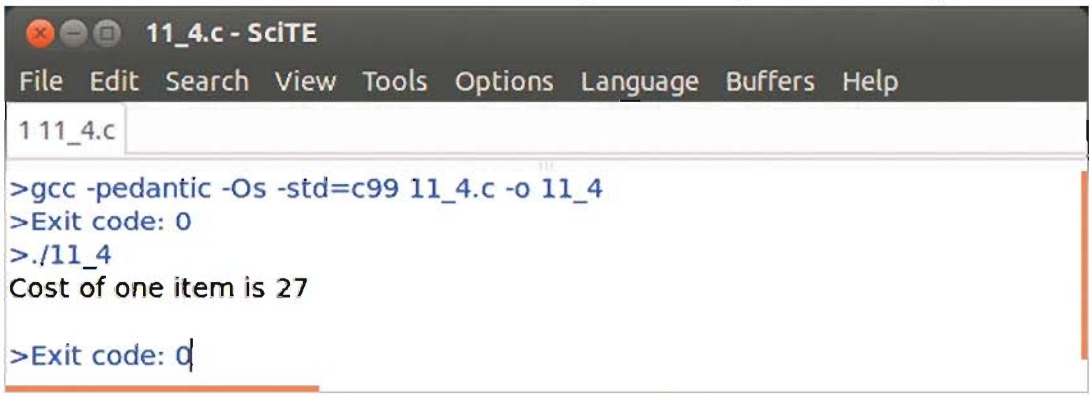
હવે, કુલ કિંમત અને વસ્તુની સંખ્યા આપવામાં આવી હોય ત્યારે વસ્તુની એકમ કિંમત શોધવા માટેનો પ્રોગ્રામ જોઈએ. પૂર્ણાંક ગણતરી રજૂ કરતો આ પ્રોગ્રામ આકૃતિ 11.8માં દર્શાવ્યો છે.



આકૃતિ 11.8 : પૂર્ણાંક ગણતરી દર્શાવતો પ્રોગ્રામ

સમજૂતી (Explanation)

અહીં પ્રથમ વિધાન દ્વારા typedef કી-વર્ડનો ઉપયોગ કરી int માટેનું ઉપનામ આપવામાં આવ્યું છે. ત્યાર પછી નવા ઉપનામનો ઉપયોગ કરી "total_cost", "quantity" અને "cost_item" નામના ત્રણ પૂર્ણાંક ચલને વ્યાખ્યાયિત કરવામાં આવ્યા છે. "total_cost" ચલમાં 650 અને "quantity" ચલમાં 24 કિંમત પણ ઉમેરવામાં આવી છે. પછીનું વિધાન "total_cost" નો "quantity" વડે ભાગાકાર કરી પરિણામનો "cost_item" ચલમાં સંગ્રહ કરે છે. ત્યાર પછી "cost_item" ચલની કિંમતને સંદેશ સાથે દર્શાવવામાં આવી છે. ઉદાહરણ 11.4નું પરિણામ આકૃતિ 11.9માં દર્શાવ્યું છે.

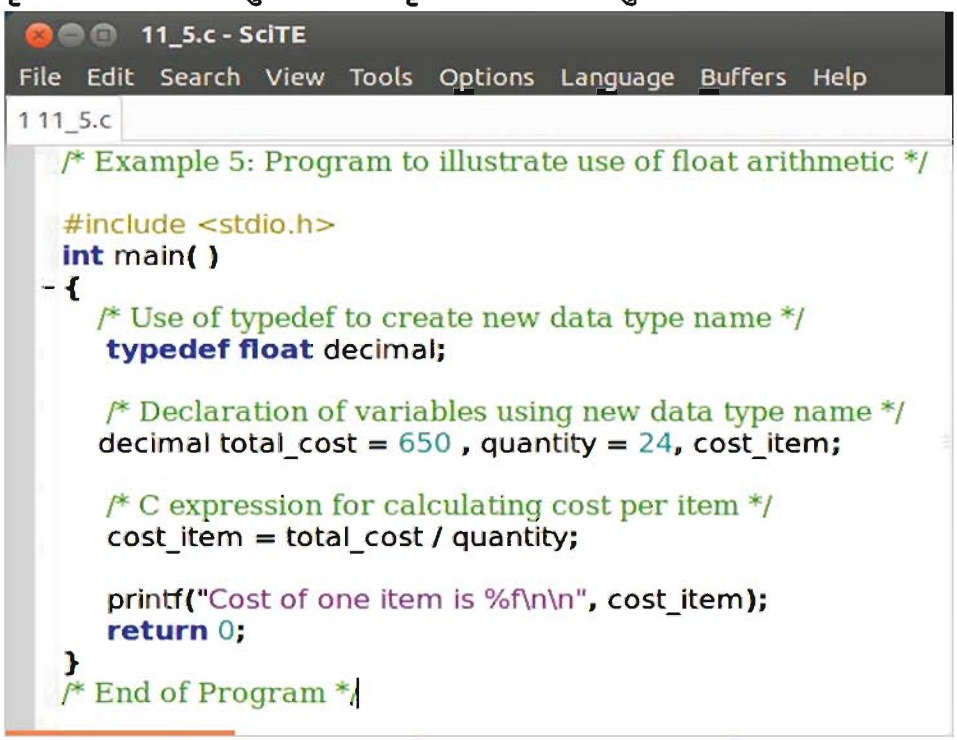


આકૃતિ 11.9 : ઉદાહરણ 11.4નું પરિણામ

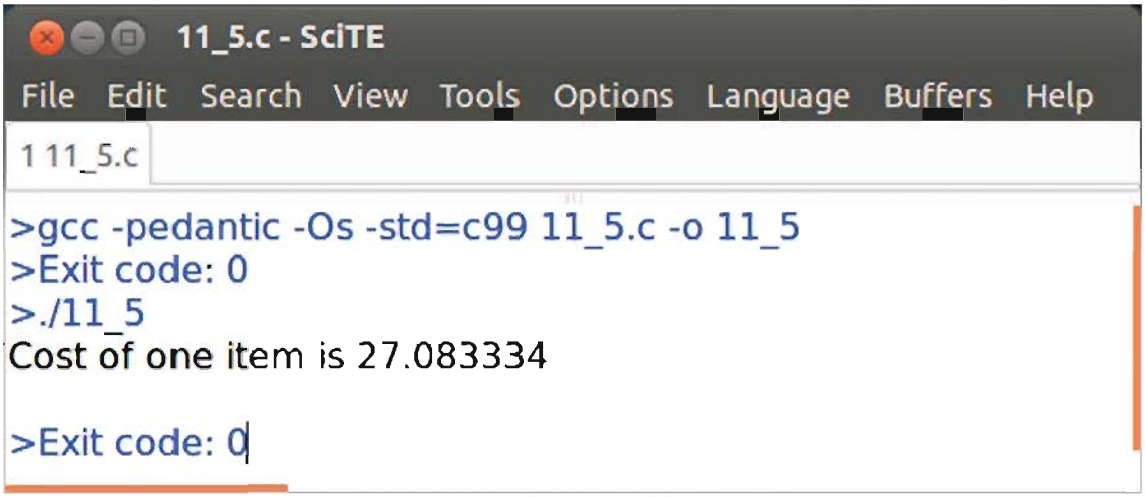
પારંપરિક ગણિતશાસ્ત્ર મુજબ 650/24નું પરિણામ 27.083333333 મળવું જોઈએ. પરંતુ પરિણામમાં "cost_item" ચલની કિંમત 27 જોઈ શકાય છે. આ ઉપરથી કહી શકાય કે પૂર્ણાંક ગણતરીનું પરિણામ હંમેશા પૂર્ણાંક સંખ્યામાં મળશે. પારંપરિક ગણિત કરતાં સી ભાષામાં ભાગાકાર માટેના પ્રક્રિયકનો ઉપયોગ અલગ છે. પારંપરિક ગણિતમાં 5/10 નું પરિણામ મળે, જે અહીં 0 મળશે.

અપૂર્ણાંક ગણતરી (Real Arithmetic)

આપેલ પદાવલિમાં ઉપયોગમાં લેવામાં આવેલા પ્રક્રિયકો જો અપૂર્ણાંક હોય તો તેને અપૂર્ણાંક ગણતરી (real arithmatic) કહે છે. અપૂર્ણાંક ગણતરીનું પરિણામ હંમેશા દશાંશચિહ્ન સાથેની કિંમતો દ્વારા દર્શાવવામાં આવે છે. પૂર્ણાંક ગણતરીને અપૂર્ણાંક ગણતરીમાં ફેરવવા માટે આકૃતિ 11.8માં દર્શાવેલ પ્રોગ્રામમાં ફેરફાર કરીએ. સુધારેલો પ્રોગ્રામ આકૃતિ 11.10માં અને તેનું પરિણામ આકૃતિ 11.11માં દર્શાવ્યું છે.



આકૃતિ 11.10 : અપૂર્ણાંક ગણતરીનું ઉદાહરણ દર્શાવતો પ્રોગ્રામ



આકૃતિ 11.11 : ઉદાહરણ 11.5નું પરિણામ

સમજૂતી (Explanation)

આ ઉદાહરણ એક અપવાદ સિવાય ઉદાહરણ 11.4 જેવું જ છે. અહીં ચલને float સ્વરૂપે ઘોષિત કર્યા છે. "total_cost" અને "quantity" ચલને આપવામાં આવેલી કિંમતો પૂર્ણાંક હોવા છતાં આંતરિક રીતે તેનો સંગ્રહ અપૂર્ણાંક (float) તરીકે થયેલ છે. માટે, પરિણામ પણ અપૂર્ણાંક સંખ્યામાં છે.

મિશ્ર પ્રકારની ગણતરી (Mixed Mode Arithmetic)

આ પ્રકારની ગણતરીમાં પૂર્ણાંક તેમજ અપૂર્ણાંક બંને પ્રકારના ઑપરેન્ડનો ઉપયોગ કરી શકાય છે. અહીં પરિણામનો આધાર ચલને આપવામાં આવેલ કિંમતોના પ્રકાર પર છે. result = 25.75 * 5; જેવી પદાવલિ સી ભાષામાં લખવી શક્ય હોવા છતાં તેનું સીધું મૂલ્યાંકન શક્ય નથી. ઉપયોગમાં લેવામાં આવેલાં તમામ ઑપરેન્ડ સમાન ડેટા પ્રકાર ધરાવતા હોય તો જ પદાવલિનું મૂલ્યાંકન શક્ય બને છે. મિશ્ર પ્રકારની ગણતરી કરી શકાતી હોવાથી સી ભાષા આ મુશ્કેલીનું નિવારણ કરે છે. સી ભાષામાં નિમ્ન સ્તરના ડેટા પ્રકારનું ઉચ્ચ ડેટા પ્રકારમાં આપોઆપ રૂપાંતરણ કરવામાં આવે છે.

નિરૂપક પ્રક્રિયકો (Assignment Operators)

અત્યાર સુધી ચર્ચેલાં ઉદાહરણોમાં આપણે total_cost = quantity * cost_item; કે date = 30; જેવાં વિધાનોનો ઉપયોગ કર્યો છે. અહીં "=" નિશાનીને નિરૂપક પ્રક્રિયક (Assignment Operator) કહે છે. કોઈ અચળ કિંમત કે પદાવલિના પરિણામનું ચલમાં નિરૂપણ કરવા માટે તેનો ઉપયોગ કરવામાં આવે છે.

સી ભાષામાં શૉર્ટહેન્ડ (short hand) પ્રક્રિયક નામે ઓળખાતાં પ્રક્રિયકો પણ પૂરાં પાડવામાં આવ્યાં છે. ગાણિતિક પ્રક્રિયકોની પાછળ "=" નિશાની ઉમેરવાથી આ પ્રક્રિયક મેળવી શકાય છે. તેનો ઉપયોગ કરવા માટેની વાક્યરચના નીચે મુજબ છે :

variable op= constant value; અથવા *variable op= expression;*

અહીં "op" એ ગાણિતિક પ્રક્રિયક છે. જ્યારે "op="ને શૉર્ટહેન્ડ પ્રક્રિયક તરીકે ઓળખવામાં આવે છે. શૉર્ટહેન્ડ પ્રક્રિયકનાં કેટલાંક ઉદાહરણ નીચે આપ્યાં છે :

first –= 1; આ વિધાન first = first – 1;ને સમકક્ષ છે.

first += 3; આ વિધાન first = first + 3;ને સમકક્ષ છે.

first *= (second + third); આ વિધાન first = first * (second + third);ને સમકક્ષ છે.

શૉર્ટહેન્ડ પ્રક્રિયક સરળ ઉપયોગિતા પૂરી પાડે છે. સરળ હોવા છતાં તેના સક્ષમ ઉપયોગ માટે થોડો મહાવરો જરૂરી છે.

સંબંધસૂચક પ્રક્રિયકો (Relational Operators)

સંબંધસૂચક પ્રક્રિયકો સમાન પ્રકારના બે ઑપરેન્ડને સરખાવવાની સુવિધા પૂરી પાડે છે અને સામાન્ય રીતે પ્રોગ્રામના અમલીકરણનો પ્રવાહ બદલવા માટે તેનો ઉપયોગ કરવામાં આવે છે. ઉદાહરણ તરીકે, આપણે ગ્રાહકને તેણે ખરીદેલી વસ્તુની રકમ તપાસી તે પ્રમાણે વળતર આપવા ઇચ્છીએ છીએ. જો કિંમત કોઈ નિશ્ચિત રકમથી વધુ હોય તો જ ગ્રાહકને વળતર આપવાનું છે. આ ઉદાહરણને સી ભાષામાં *if (total_purchase > 10000) discount = 500;* તરીકે રજૂ કરી શકાય.

સરખામણી કરવા માટે સી ભાષા છ સંબંધસૂચક પ્રક્રિયકો પૂરાં પાડે છે. તમામ સંબંધસૂચક પ્રક્રિયકોની યાદી અને તેના ઉપયોગો કોષ્ટક 11.5માં દર્શાવ્યા છે.

પ્રક્રિયક	ઉપયોગ
= =	બે ઑપરેન્ડની સમાનતા (equality) ચકાસશે.
! =	બે ઑપરેન્ડની અસમાનતા (non equality) ચકાસશે.
>	બે ઑપરેન્ડમાંથી મોટી (greater) કિંમત ચકાસશે.
<	બે ઑપરેન્ડમાંથી નાની (smaller) કિંમત ચકાસશે.
>=	બે ઑપરેન્ડમાંથી મોટી કિંમત અથવા સમાનતા (greater or equality) ચકાસશે.
<=	બે ઑપરેન્ડ માટે નાની કિંમત અથવા સમાનતા (smaller or equality) ચકાસશે.

કોષ્ટક 11.5 : સંબંધસૂચક પ્રક્રિયકો

અહીં એ નોંધવું જરૂરી છે કે બે ઑપરેન્ડની સમાનતા ચકાસવા માટે "="ને બદલે "= =" પ્રક્રિયકનો ઉપયોગ થાય છે. કારણ કે, સી ભાષામાં "=" પ્રક્રિયકનો ઉપયોગ નિરૂપક પ્રક્રિયક તરીકે કરવામાં આવે છે. સંબંધસૂચક પ્રક્રિયકોના ઉપયોગ માટે નીચેની વાક્યરચનાનો ઉપયોગ કરવામાં આવે છે :

expression-1 Rop expression-2

અહીં "Rop" એટલે સંબંધસૂચક પ્રક્રિયક (relational operator) તથા expression-1 અને expression-2 ગાણિતિક પદાવલિ, ચલ કે અચળ હોઈ શકે છે. "if" જેવા નિર્ણય માળખાં તથા "for", "while" અને "do..while" જેવા નિયંત્રણ માળખાં સાથે સંબંધસૂચક પ્રક્રિયકોનો ઉપયોગ કરવામાં આવે છે. આ માળખાઓ વિશે પ્રકરણ-13 અને 14માં વિસ્તૃત ચર્ચા કરવામાં આવી છે. સંબંધસૂચક પ્રક્રિયકનો ઉપયોગ કર્યો હોય તેવા ઘણા પ્રોગ્રામ આ પુસ્તકમાં જોઈ શકાશે.

વધારા / ઘટાડા સૂચક પ્રક્રિયકો (Increment / Decrement Operators)

પ્રોગ્રામમાં આવેલ ચલની કિંમતમાં કેટલીકવાર 1 જેટલો વધારો કે ઘટાડો કરવાની જરૂર પડતી હોય છે. સામાન્ય રીતે વિધાનોના ગણને પુનરાવર્તિત કરવાના હોય ત્યારે આ પ્રકારની સ્થિતિ ઉદ્ભવે છે. અહીં લૂપના અમલની સંખ્યાની દેખરેખ રાખે એવા ચલની જરૂર પડે છે. આ માટે શૉર્ટહૅન્ડ પ્રક્રિયકો (short hand operators)નો ઉપયોગ કરી શકાય છે. ઉદાહરણ તરીકે આ માટે variable += 1 અથવા variable – = 1 વાક્યરચનાનો ઉપયોગ કરી શકાય.

સી ભાષા આ કાર્ય માટે બે વિશિષ્ટ યુનરી (unary) પ્રક્રિયકો "++" અને "– –" પૂરાં પાડે છે. આ પ્રક્રિયકોને એક જ ઑપરેન્ડની જરૂર પડે છે. "++"ને વધારા સૂચક (Increment) પ્રક્રિયક અને "– –" ને ઘટાડા સૂચક (Decrement) પ્રક્રિયક તરીકે ઓળખવામાં આવે છે. આ પ્રક્રિયકોના ઉપયોગ માટેની વાક્યરચના નીચે મુજબ છે :

++ variable; અથવા variable ++

– – variable; અથવા variable – –

અહીં, "++" અને "– –"નો ઉપયોગ ચલની આગળ કે પાછળ ઉમેરીને કરવામાં આવે છે. ચલની આગળ ઉમેરવામાં આવે ત્યારે તેને 'પ્રી-ઇન્ક્રિમેન્ટ' (pre-increment) અથવા પ્રી-ડિક્રિમેન્ટ (pre-decrement) પ્રક્રિયક તરીકે ઓળખવામાં આવે છે. અન્યથા તેને ચલની પાછળ ઉમેરવામાં આવે તો પોસ્ટ ઇન્ક્રિમેન્ટ (post increment) અથવા પોસ્ટ ડિક્રિમેન્ટ (post decrement) પ્રક્રિયક કહેવાય છે.

આ પ્રક્રિયકોના પરિણામ સ્વરૂપે ચલની કિંમતમાં 1નો વધારો કે ઘટાડો કરવામાં આવે છે. આ વધારા કે ઘટાડાની અસર પ્રોગ્રામમાં કરવામાં આવેલ પ્રક્રિયકની રીત પર અવલંબે છે. ધારો કે વિધાન નીચે મુજબ લખ્યાં છે :

int first = 15, second = 20, result;

result = first + second++;

અહીં, "result" ચલની કિંમત 35 થશે, 36 નહીં. જ્યારે બીજા વિધાનને અંતે "second" ચલની કિંમત 21 બનશે. તેની પહેલાંની કિંમતમાં 1 ઉમેરવામાં આવશે.

અહીં એ નોંધવું જરૂરી છે કે પોસ્ટ ઇન્ક્રીમેન્ટ પ્રક્રિયકનો ઉપયોગ કરવામાં આવે ત્યારે પદાવલિનું મૂલ્યાંકન કરવા માટે ચલની જૂની કિંમતનો ઉપયોગ કરવામાં આવેશે અને ત્યાર પછી ચલની કિંમતમાં 1 નો વધારો કરવામાં આવે છે.

આ જ રીતે જો નીચેનાં વિધાનો લખવામાં આવે,

int first = 15, second = 20, result;

result = first + ++second;

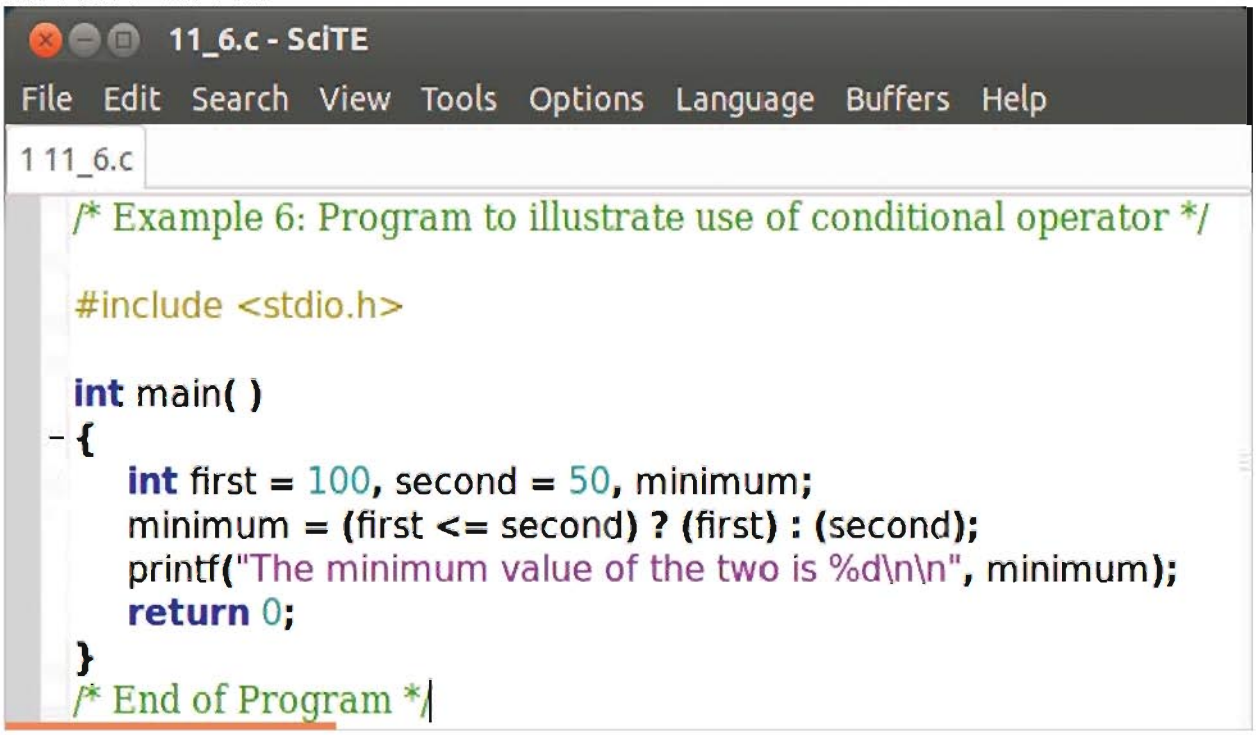
તો "result" ચલની પરિણામી કિંમત 36 હશે. જ્યારે બીજા વિધાનના અંતે "second" ચલની કિંમત 21 બનશે. અહીં એ નોંધવું જરૂરી છે કે, પ્રી-ઇન્ક્રિમેન્ટ પ્રક્રિયકનો ઉપયોગ કરવામાં આવે ત્યારે સૌ પ્રથમ ચલની કિંમતમાં 1નો વધારો કરવામાં આવે છે તથા ત્યાર પછી આ નવી કિંમતનો ઉપયોગ કરી પદાવલિનું મૂલ્યાંકન કરવામાં આવે છે.

શરતી પ્રક્રિયક (Conditional Operators)

શરતોને ચકાસવા માટે સી ભાષા ટર્નરી પ્રક્રિયક (ternary operator) અથવા શરતી પ્રક્રિયક (conditional operator) નામે ઓળખાતાં પ્રક્રિયક પૂરાં પાડે છે. આ પ્રક્રિયકને વ્યાખ્યાયિત કરવા માટે બે નિશાનીઓ- "? :" નો ઉપયોગ કરવામાં આવે છે. શરતી પ્રક્રિયકના ઉપયોગ માટેની વાક્યરચના નીચે મુજબ છે :

(condition) ? (True statement) : (False statement);

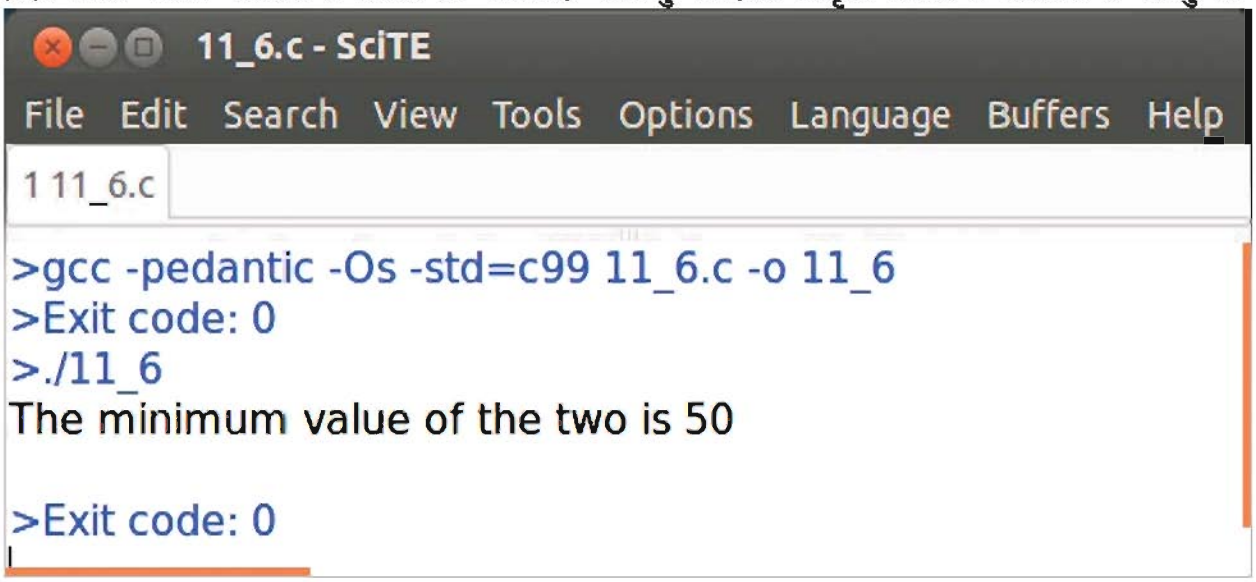
અહીં, condition એ first < second જેવી કોઈ સંબંધસૂચક પદાવલિનો નિર્દેશ કરે છે. બે સંખ્યાઓમાંથી નાની સંખ્યા શોધવા માટેના પ્રોગ્રામની મદદથી શરતી પ્રક્રિયકનો ઉપયોગ સમજીએ. આકૃતિ 11.12 આ પ્રોગ્રામ માટેનું કોડ લિસ્ટિંગ દર્શાવે છે.



આકૃતિ 11.12 : શરતી પ્રક્રિયકનો ઉપયોગ દર્શાવતો પ્રોગ્રામ

સમજૂતી (Explanation)

પ્રોગ્રામનું પ્રથમ વિધાન ત્રણ પૂર્ણાંક ચલની ઘોષણા કરી તેમાંથી બે ચલમાં કિંમત ઉમેરે છે. "first" અને "second" ચલમાં અનુક્રમે 100 અને 50 કિંમતો ઉમેરવામાં આવે છે. ત્યાર પછી શરતી પ્રક્રિયક દ્વારા ચકાસવામાં આવે છે કે "first" ચલની કિંમત "second" ચલની કિંમત કરતાં ઓછી અથવા તેના જેટલી છે કે નહીં. જો શરત સાચી (true) હોય તો "minimum" ચલને first ચલની કિંમત આપવામાં આવે છે. અન્યથા તેમાં 'second' ચલની કિંમત ઉમેરવામાં આવે છે. ત્યાર પછી યોગ્ય સંદેશ સાથે minimum ચલની કિંમત સ્ક્રીન પર દર્શાવવામાં આવે છે. અંતમાં પ્રોગ્રામમાંથી બહાર નીકળવાનો નિર્દેશ છે. ઉદાહરણ 11.6નું પરિણામ આકૃતિ 11.13માં દર્શાવવામાં આવ્યું છે.



આકૃતિ 11.13 : ઉદાહરણ 11.6નું પરિણામ

તાર્કિક પ્રક્રિયકો (Logical Operators)

કેટલીકવાર પરિણામ મેળવવા માટે એક જ સમયે એકથી વધુ શરતોની ચકાસણી કરવાની જરૂર પડે છે. જ્યારે કેટલીકવાર શરતોના ગણમાંથી કોઈ પણ એક શરત સંતોષાય તો પણ પરિણામ મેળવવું જરૂરી બને છે. આ પ્રકારના સંબંધને સામાન્ય રીતે તાર્કિક સંબંધ (logical relation) તરીકે ઓળખવામાં આવે છે. ઉદાહરણ તરીકે, બે વિષયોની પરીક્ષા આપી હોય તેવા કોઈ વિદ્યાર્થીનો વિચાર કરો. જો તે વિદ્યાર્થીએ બંને વિષયોમાં 35 કે તેથી વધુ ગુણ મેળવ્યા હશે તો જ તેને 'ઉત્તીર્ણ' જાહેર કરવામાં આવશે. અહીં આ સંબંધને 'જો (પ્રથમ વિષયના ગુણ >= 35) અને (બીજા વિષયના ગુણ >= 35) હોય તો ઉત્તીર્ણ અન્યથા અનુત્તીર્ણ'- આ પ્રકારે દર્શાવી શકાય.

આ પ્રકારના સંબંધ રજૂ કરવા માટે સી ભાષા તાર્કિક પ્રક્રિયકોની સુવિધા પૂરી પાડે છે. કોષ્ટક 11.6 આ તાર્કિક પ્રક્રિયકોની યાદી અને તેના ઉપયોગો દર્શાવે છે.

પ્રક્રિયક	ઉપયોગ
&&	તાર્કિક AND
||	તાર્કિક OR
!	તાર્કિક NOT

કોષ્ટક 11.6 : તાર્કિક પ્રક્રિયકો

આપેલ તમામ શરતો ફરજિયાત સંતોષવી હોય ત્યારે AND તાર્કિક પ્રક્રિયકનો ઉપયોગ કરવામાં આવે છે. તથા આપેલ તમામ શરતોમાંથી ઓછામાં ઓછી એક શરત સંતોષવી હોય ત્યારે OR તાર્કિક પ્રક્રિયકનો ઉપયોગ કરવામાં આવે છે. તાર્કિક પ્રક્રિયકોના પરિણામ સ્વરૂપે 0 અથવા 1 મળે છે. અહીં 'શૂન્ય' એ 'ખોટા' (false) તથા 1 એ 'સાચા' (true) પરિણામનો નિર્દેશ કરે છે. કોષ્ટક 11.7 જુદા જુદા તાર્કિક પ્રક્રિયકોનું પરિણામ દર્શાવે છે.

Expression 1	Expression 2	! (Expression 2)	Expression 1 && Expression 2	Expression 1 || Expression 2
0	0	1	0	0
0	1	1	0	1
1	0	0	0	1
1	1	0	1	1

કોષ્ટક 11.7 : તાર્કિક પ્રક્રિયકોનું પરિણામ

અહીં જોઈ શકાય છે કે, જો બંને પદાવલિ '1' પરત કરે તો જ AND પ્રક્રિયક 1 પરત કરે છે. જ્યારે આપેલમાંથી કોઈ પણ એક પદાવલિ 1 પરત કરે તો OR પ્રક્રિયક 1 પરત કરે છે.

હવે, નીચે આપેલ માપદંડને આધારે વિદ્યાર્થીનો વર્ગ (Grade) શોધી આપે તે માટેનો પ્રોગ્રામ બનાવીએ :

વર્ગ (Grade)	ગુણ (Marks)
A વર્ગ	જો ટકા 70 કે તેથી વધુ હોય
B વર્ગ	જો ટકા 60 થી 69ની વચ્ચે હોય
C વર્ગ	જો ટકા 35 થી 59ની વચ્ચે હોય
F વર્ગ	જો ટકા 35 થી ઓછા હોય.

આકૃતિ 11.14માં આ પ્રોગ્રામનું કોડ લિસ્ટિંગ આપવામાં આવ્યું છે તથા આકૃતિ 11.15 તેનું પરિણામ દર્શાવે છે.

11_7.c - SciTE

File Edit Search View Tools Options Language Buffers Help

11_7.c

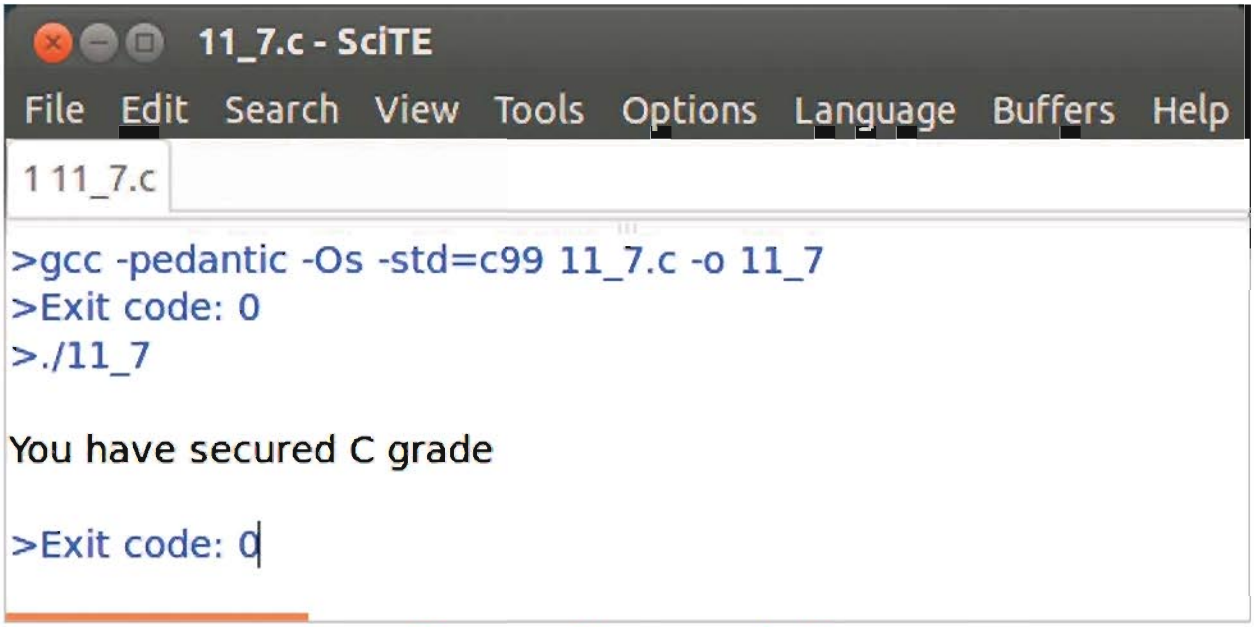
```c
/* Example 7: Program to illustrate use of logical operator */

#include <stdio.h>
int main( )
{
    int percentage = 55;

    /* Verify whether percentage is between 0 and 100 using AND */
    if ( percentage >= 0 && percentage <= 100 )
    {
      /* Check whether percentage greater than 70 */
      if ( percentage >= 70 )
      printf("\nYou have secured A grade\n\n");

      /* Check whether  percentage between 60 and 69 */
      if ( percentage >= 60 && percentage < 70 )
      printf("\nYou have secured B grade\n\n");

      /* Check whether  percentage between 35 and 60 */
      if ( percentage >= 35 && percentage < 60 )
      printf("\nYou have secured C grade\n\n");

      /* Check whether percentage less than 35 */
     if ( percentage < 35 )
      printf("\nYou have secured F grade\n\n");
    }
    else    /* Executed when percentage below 0 or above 100 */
    {
      printf("\nYou have entered wrong value in percentage");
      printf("\nPlease enter value between 0 and 100\n\n");
    }
    return 0;
}
/* End of Program */
```

આકૃતિ 11.14 : તાર્કિક પ્રક્રિયકનો ઉપયોગ દર્શાવતો પ્રોગ્રામ

સમજૂતી (Explanation)

અહીં પ્રથમ વિધાન દ્વારા "percentage" નામના પૂર્ણાંક ચલને ઘોષિત કરી તેમાં કિંમત આપવામાં આવે છે. શરૂઆતમાં "percentage" ચલની કિંમત 0 થી 100 વચ્ચે આપેલ છે કે નહીં તે ચકાસવામાં આવે છે. જો જવાબ 'ના' મળે તો 'else' વિભાગમાં આવેલ સંદેશ "You have entered wrong value in percentage, Please enter value between 0 and 100" દર્શાવી પ્રોગ્રામમાંથી બહાર નીકળી શકાય છે.

જો સરખામણીનું પરિણામ 'હા' (true) મળશે તો ટકાની કિંમતને if વિધાન અને તાર્કિક પ્રક્રિયકો દ્વારા ચકાસવામાં આવશે. જો if વિધાનમાં આપેલ માપદંડનું પરિણામ હકારાત્મક મળશે તો તેનો સંદેશ સ્ક્રીન પર દર્શાવવામાં આવશે. આકૃતિ 11.15 જુઓ.



આકૃતિ 11.15 : ઉદાહરણ 11.7નું પરિણામ

બિટવાઇઝ પ્રક્રિયકો (Bitwise Operators)

આપણે આ પ્રકરણમાં જોયું કે ડેટાને મેમરીસ્થાન પર બિટ (bit) સ્વરૂપે સંગ્રહવામાં આવે છે. સીધા જ બિટ સ્તરે કાર્ય કરવા માટે સી ભાષામાં બિટવાઇઝ પ્રક્રિયકોની સુવિધા આપવામાં આવી છે. સી ભાષામાં ઉપલબ્ધ બિટવાઇઝ પ્રક્રિયકોની યાદી અને તેના ઉપયોગ કોષ્ટક 11.8માં આપવામાં આવ્યા છે.

પ્રક્રિયક	ઉપયોગ
&	Bitwise AND
|	Bitwise OR
~	Bitwise NOT
^	Bitwise Exclusive OR
<<	આપેલ બિટ જેટલા અંકોને ડાબી બાજુ ખસેડવા
>>	આપેલ બિટ જેટલા અંકોને જમણી બાજુ ખસેડવા

કોષ્ટક 11.8 : બિટવાઇઝ પ્રક્રિયકો (Bitwise Operators)

બિટવાઇઝ AND, OR, Exclusive OR (XOR), Left Shift અને Right Shift પ્રક્રિયકોનો ઉપયોગ કરવા માટે બે ઑપરેન્ડ જરૂરી છે. બીજી તરફ NOT પ્રક્રિયક માત્ર એક ઑપરેન્ડ સાથે કાર્ય કરે છે. અહીં ઑપરેન્ડ 0 અને 1 કિંમતો સ્વરૂપે હોય છે. આ પ્રક્રિયકોની વિસ્તૃત ચર્ચા પુસ્તકની મર્યાદા બહાર છે.

વિશિષ્ટ પ્રક્રિયકો (Special Operators)

સી ભાષા sizeof( ), ",", ".", "–>", "&" અને "*" જેવાં વિશિષ્ટ પ્રક્રિયકો પૂરાં પાડે છે. આ વિભાગમાં આપણે માત્ર "," અને sizeof() પ્રક્રિયકોની ચર્ચા કરીશું. "," (અલ્પવિરામ) પ્રક્રિયકનો ઉપયોગ ઘણી જગ્યાએ કરવામાં આવે છે. અલ્પવિરામ (comma) પ્રક્રિયકનો ઉપયોગ નિર્ણય માળખાં અને નિયંત્રણ માળખાંમાં પણ કરવામાં આવે છે જેની ચર્ચા હવે પછીના પ્રકરણોમાં કરવામાં આવી છે.

ઘટકનો સંગ્રહ કરવા માટે જરૂરી બાઇટની સંખ્યા પરત કરવા માટે sizeof( ) નામના એક વિશિષ્ટ પ્રક્રિયકનો ઉપયોગ કરવામાં આવે છે. ઉદાહરણ તરીકે, size = *sizeof(int)* વિધાન "size" ચલમાં "4" કિંમતનો સંગ્રહ કરશે. કારણ કે int ડેટા પ્રકાર મેમરીની 4 બાઇટ જેટલી જગ્યાનો ઉપયોગ કરે છે.

અલ્પવિરામ (comma) પ્રક્રિયક અને sizeof( ) પ્રક્રિયકનો ઉપયોગ દર્શાવતો પ્રોગ્રામ આકૃતિ 11.16માં આપેલ છે તથા તેનું પરિણામ 11.17માં દર્શાવવામાં આવ્યું છે.

11_8.c - SciTE

File Edit Search View Tools Options Language Buffers Help

1 11_8.c

```c
/* Example 8: Program to illustrate use of comma and size of operator */

#include <stdio.h>
int main( )
{
    float first, second, result;
    int  size;
    /* Use of comma operator */
    result = (first = 125, second = 15, first / second);
    printf("\nFirst = %f, Second = %f, Result = %f", first, second, result);

    /* Use of sizeof operator */
    size = sizeof(result);
    printf("\nSize allocated by compiler to variable result is %d bytes\n\n", size);

}
/* End of Program */
```

આકૃતિ 11.16 : વિશિષ્ટ પ્રક્રિયકોનો ઉપયોગ દર્શાવતો પ્રોગ્રામ

11_8.c - SciTE

File Edit Search View Tools Options Language Buffers Help

1 11_8.c

```
>gcc -pedantic -Os -std=c99 11_8.c -o 11_8
>Exit code: 0
>./11_8

First = 125.000000, Second = 15.000000, Result = 8.333333
Size allocated by compiler to variable result is 4 bytes

>Exit code: 0
```

આકૃતિ 11.17 : ઉદાહરણ 11.8નું પરિણામ

સમજૂતી (Explanation)

અહીં, પ્રથમ વિધાન ત્રણ અપૂર્ણ (float) ચલની ઘોષણા કરે છે તથા બીજું વિધાન એક પૂર્ણાંક ચલને ઘોષિત કરે છે. અમલ કરી શકાય તેવા ત્રીજા વિધાનમાં અલ્પવિરામ (comma) પ્રક્રિયકનો ઉપયોગ કરી પદાવલિ લખવામાં આવી છે. આપેલ વિધાન આ મુજબ કાર્ય કરશે : પ્રથમ જમણી બાજુની પદાવલિનું મૂલ્યાંકન કરવામાં આવશે. પ્રથમ પ્રક્રિયામાં "first" ચલને 125 કિંમત આપવામાં આવશે, બીજી પ્રક્રિયામાં "second" ચલને 15 કિંમત આપવામાં આવશે. અને ત્રીજી પ્રક્રિયામાં "first" નો "second" વડે ભાગાકાર કરવામાં આવશે. અંતમાં, ભાગાકારનું પરિણામ "result" ચલમાં સંગ્રહવામાં આવશે. ત્યાર પછીનું વિધાન "first", "second" અને "result" ચલની કિંમત દર્શાવે છે. ત્યાર પછીના વિધાનમાં "result" ચલમાં આવેલ કિંમતને મેમરીમાં આપવામાં આવેલી જગ્યાનો "size" નામના ચલમાં સંગ્રહ કરવામાં આવશે.

પદાવલિનું મૂલ્યાંકન (Evaluation of Expression)

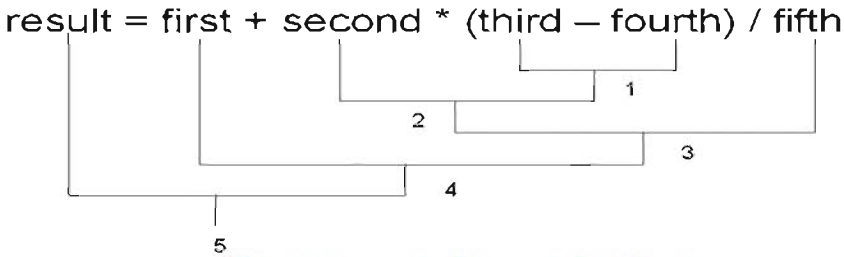
અત્યાર સુધીમાં આપણે સી ભાષામાં ઉપલબ્ધ પ્રક્રિયકોનો અભ્યાસ કરી તેનો પદાવલિમાં કેવી રીતે ઉપયોગ કરવો તે શીખ્યા. હવે, પદાવલિનું મૂલ્યાંકન કેવી રીતે કરવામાં આવે છે તે જોઈએ. નીચેના વિધાનમાં રહેલ પદાવલિ જુઓ :

*result = first + second * third – fourth;*

અહીં, પ્રથમ 'second' અને 'third' ચલની કિંમતોનો ગુણાકાર કરવામાં આવશે. આ ગુણાકારની કિંમતને ત્યાર પછી first ચલની કિંમતમાં ઉમેરવામાં આવશે અને આ મધ્યવર્તી કિંમતમાંથી fourthની કિંમત બાદ કરવામાં આવશે. જો પદાવલિમાં એક જ અગ્રતા ધરાવતા બે પ્રક્રિયકોનો ઉપયોગ કરવામાં આવ્યો હશે તો તેમનું મૂલ્યાંકન ડાબીથી જમણી તરફ કરવામાં આવશે. કૌંસનો ઉપયોગ કરી આ ક્રમને બદલી શકાય છે.

*result = first + second * (third – fourth) / fifth;*

હવે અહીં third - fourth નું મૂલ્યાંકન સૌપ્રથમ કરવામાં આવશે, મધ્યવર્તી કિંમતનો second ચલની કિંમત સાથે ગુણાકાર કરવામાં આવશે, પછી fifth ચલની કિંમત વડે ભાગાકાર કરવામાં આવશે અને અંતમાં first ચલની કિંમતમાં તેને ઉમેરવામાં આવશે. આકૃતિ 11.18 જુઓ.



આકૃતિ 11.18 : પદાવલિના મૂલ્યાંકનનો ક્રમ

પ્રક્રિયકોની અગ્રતા (Priority of Operators)

અત્યાર સુધી જોયેલા પ્રક્રિયકોને તેમની નિશ્ચિત અગ્રતા આપવામાં આવી છે. ઉદાહરણ result = first + second * (third – fourth) / fifth; વિધાનમાં જોઈ શકાય છે કે * ને +, – કે / કરતાં વધુ અગ્રતા આપવામાં આવી હતી. સી ભાષાના સર્જકોએ આ અગ્રતા પૂર્વનિશ્ચિત કરેલી છે. આમાંના કેટલાંક પ્રક્રિયકોની અગ્રતા કોષ્ટક 11.9માં દર્શાવી છે. પરિશિષ્ટ III માં આ માટેની વિસ્તૃત સમજ આપવામાં આવી છે.

Operator	Operation Used for	Associativity	Precedence
( )	Function call	Left to Right	First
[ ]	Array expression		
++	Increment	Right to Left	Second
- -	Decrement		
sizeof( )	Size of operand		
*	Multiplication	Left to Right	Third
/	Division		
%	Modulo division		
+	Binary addition	Left to Right	Fourth
-	Binary subtraction		

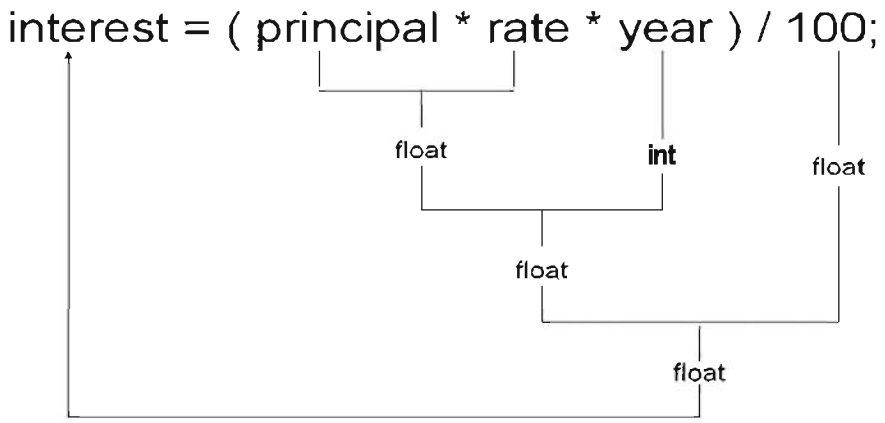
કોષ્ટક 11.9 : પ્રક્રિયકોની અગ્રતા

પ્રકાર બદલી (Type Conversion)

પદાવલિમાં આવેલ તમામ ઓપરેન્ડ એક જ પ્રકારના હોય તો જ તેનું મૂલ્યાંકન કરવામાં આવે છે. આ લાક્ષણિકતાને કારણે પદાવલિના મૂલ્યાંકન માટે તેમાં આવેલા ડેટા પ્રકારની આંતરિક બદલી જરૂરી બને છે. ઉદાહરણ તરીકે નીચેનો કોડ જુઓ :

```
int year=2;
float principal, rate, interest;
interest = (principal * rate * year) / 100;
```

આ પદાવલિનું મૂલ્યાંકન આકૃતિ 11.19માં દર્શાવ્યા મુજબ કરવામાં આવશે :

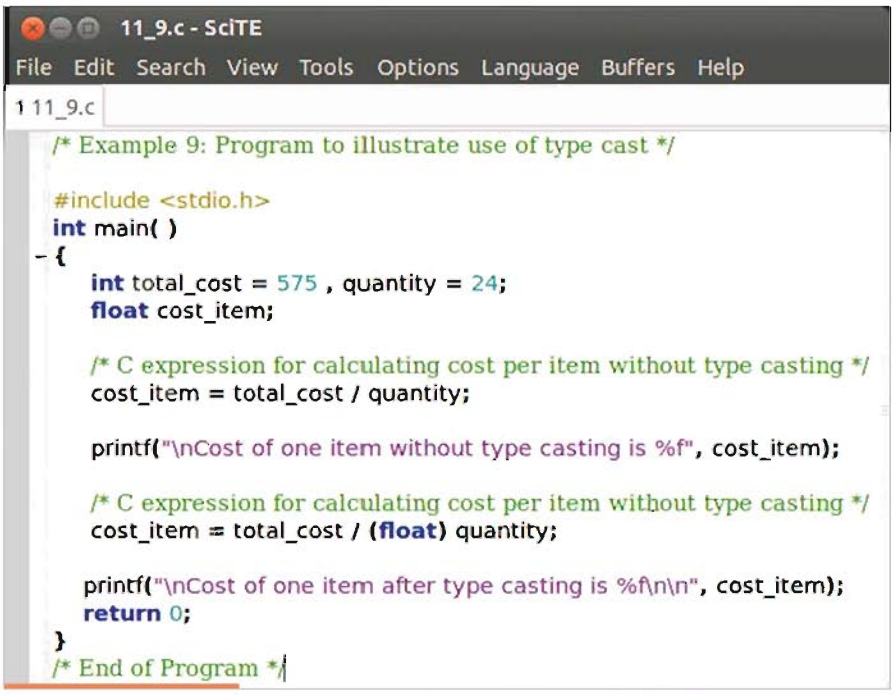


આકૃતિ 11.19 : પ્રકાર બદલી

આંતરિક ફેરબદલીને ઉપરવટ (override) થવા માટે ટાઇપ કાસ્ટિંગ (type casting) નામની પ્રક્રિયાનો ઉપયોગ કરી શકાય છે. કોઈ પણ કિંમતનો પ્રકાર બદલવા (type cast) માટે નીચે આપેલ વાક્યરચનાનો ઉપયોગ કરવામાં આવે છે.

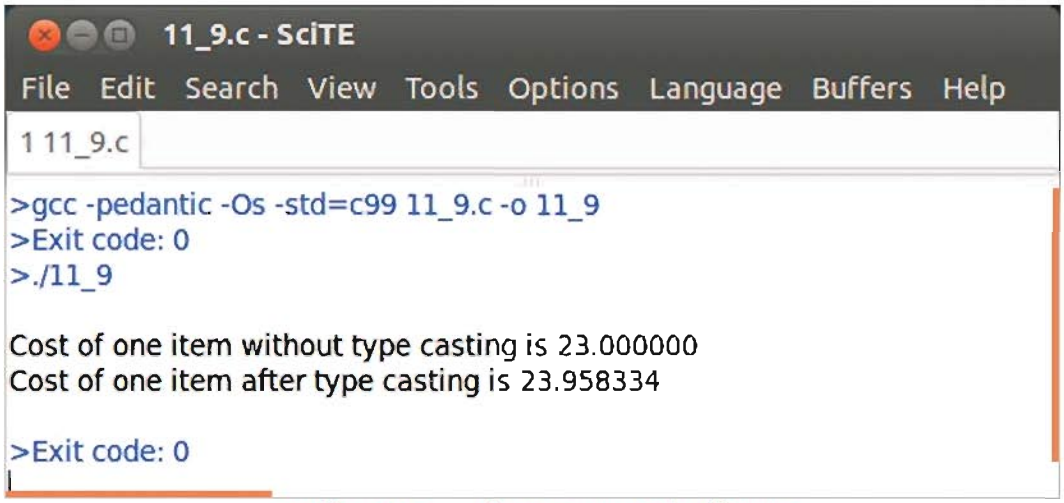
(data type) variable અથવા *(datatype) constant*

ટાઇપ કાસ્ટિંગનો ઉપયોગ કરી બનાવવામાં આવેલા પ્રોગ્રામનું કોડ લિસ્ટિંગ આકૃતિ 11.20 માં આપેલ છે. આકૃતિ 11.21 પ્રોગ્રામનું પરિણામ દર્શાવે છે.




```c
/* Example 9: Program to illustrate use of type cast */

#include <stdio.h>
int main( )
{
    int total_cost = 575 , quantity = 24;
    float cost_item;

    /* C expression for calculating cost per item without type casting */
    cost_item = total_cost / quantity;

    printf("\nCost of one item without type casting is %f", cost_item);

    /* C expression for calculating cost per item without type casting */
    cost_item = total_cost / (float) quantity;

    printf("\nCost of one item after type casting is %f\n\n", cost_item);
    return 0;
}
/* End of Program */
```

આકૃતિ 11.20 : ટાઇપ કાસ્ટિંગ દર્શાવતો પ્રોગ્રામ



આકૃતિ 11.21 : ઉદાહરણ 11.9નું પરિણામ

સમજૂતી (Explanation)

આ પ્રોગ્રામમાં 24 વસ્તુઓની કિંમત આપેલી છે અને વસ્તુની એકમ કિંમત શોધવાની છે. "total_cost" અને "quantity" ચલને પૂર્ણાંક તથા "cost_item" ચલને અપૂર્ણાંક તરીકે વ્યાખ્યાયિત કરવામાં આવ્યા છે. એ જોઈ શકાય છે કે બંને ઑપરેન્ડ પૂર્ણાંક હોવાને કારણે પ્રથમ સી પદાવલિ વસ્તુની એકમ કિંમત તરીકે 23.000000 દર્શાવે છે. આ જ પદાવલિનો જ્યારે ટાઇપ કાસ્ટિંગ સાથે ઉપયોગ કરવામાં આવે છે ત્યારે તે ઇચ્છિત પરિણામ 23.958334 દર્શાવે છે.

સંગ્રાહક વર્ગો (Storage Classes)

સી પ્રોગ્રામમાં ઉપયોગમાં લેવામાં આવતા ચલનો સંગ્રહ સામાન્ય રીતે કમ્પ્યૂટરની પ્રાઇમરી મેમરીમાં કરવામાં આવે છે. કેટલાક ચલનો રજિસ્ટરમાં સંગ્રહ કરવો પણ શક્ય છે. સી ભાષા automatic, external, register, અને static નામના ચાર સંગ્રાહક વર્ગો (storage classes) પૂરાં પાડે છે. આ સંગ્રાહક વર્ગો દ્વારા ચલનો સંગ્રહ કરવા માટેનું સ્થાન નિશ્ચિત કરવામાં આવે છે. તમામ સંગ્રાહક વર્ગોની એક પછી એક ચર્ચા કરીએ.

ઑટોમેટિક ચલ (Automatic Variables)

તમામ ચલનો પૂર્વ નિર્ધારિત સંગ્રાહક વર્ગ 'ઑટોમેટિક' હોય છે. ચલને 'ઑટોમેટિક' સ્વરૂપમાં નિશ્ચિત રૂપે (explicitely) વ્યાખ્યાયિત કરવા માટે auto કી-વર્ડનો ઉપયોગ કરવામાં આવે છે. ચલ માટે ઓટોમેટિક સંગ્રાહક વર્ગ વ્યાખ્યાયિત કરવા માટેની વાક્યરચના નીચે મુજબ છે :

auto data type identifier;

ઉદાહરણ તરીકે, auto int number; વ્યાખ્યા દર્શાવે છે કે ચલ number એ પૂર્ણ પ્રકારનો છે અને તેનો સંગ્રાહક વર્ગ ઓટોમેટિક છે. જ્યારે કોઈ વિધેયનો અમલ કરવામાં આવે છે (called) ત્યારે આ પ્રકારના ચલની રચના કરવામાં આવે છે તથા જ્યારે આ વિધેય બોલાવનાર (caller) વિધેયને નિયંત્રણ પરત મોકલે છે ત્યારે આ પ્રકારના ચલનો નાશ કરવામાં આવે છે. આ પ્રકારના ચલને શરૂઆતમાં પૂર્વનિર્ધારિત રીતે અનિશ્ચિત કિંમત (garbage value - એવી કિંમત કે ઉપયોગકર્તા માટે જેનું કોઈ મહત્ત્વ નથી) આપવામાં આવે છે અને તેનો પ્રાઇમરી મેમરીમાં સંગ્રહ કરવામાં આવે છે.

એક્સ્ટર્નલ ચલ (External Variable)

કેટલીકવાર પ્રોગ્રામરને કોઈ ચલ બે વિધેય કે પ્રોગ્રામ વચ્ચે વહેંચવાની જરૂર પડે છે. બે જુદા જુદા વિધેય કે પ્રોગ્રામ વચ્ચે ચલની વહેંચણી માટે 'એક્સ્ટર્નલ' સંગ્રાહક વર્ગનો ઉપયોગ કરી શકાય છે. આ માટે એ જરૂરી છે કે ઓછામાં ઓછા એક પ્રોગ્રામમાં તે ચલ વૈશ્વિક (global) ચલ તરીકે ઘોષિત થયેલો હોવો જોઈએ. નીચેની વાક્યરચના દ્વારા ચલ માટે એક્સ્ટર્નલ સંગ્રાહક વર્ગ વ્યાખ્યાયિત કરી શકાય.

extern datatype identifier;

ઉદાહરણરૂપે, extern char choice; વ્યાખ્યા દર્શાવે છે કે choice ચલ અક્ષર પ્રકારનો છે અને તેનો સંગ્રાહક વર્ગ એક્સ્ટર્નલ છે. એક્સ્ટર્નલ ચલની પૂર્વ નિર્ધારિત કિંમત શૂન્ય છે અને તેનો પ્રાઇમરી મેમરીમાં સંગ્રહ કરવામાં આવે છે.

રજિસ્ટર ચલ (Register Variable)

ઝડપી ઉપયોગ કરવા માટે ચલને CPU રજિસ્ટરમાં સંગ્રહી શકાય છે. સી ભાષા આ માટે 'register' નામનો સંગ્રાહક વર્ગ પૂરો પાડે છે. નીચેની વાક્યરચના દ્વારા ચલ માટે રજિસ્ટર સંગ્રાહક વર્ગ વ્યાખ્યાયિત કરી શકાય છે :

register datatype identifier;

ઉદાહરણ રૂપે, register int counter; વ્યાખ્યા દર્શાવે છે કે counter ચલ પૂર્ણાંક પ્રકારનો છે અને તેનો સંગ્રાહક વર્ગ રજિસ્ટર છે. પૂર્વ નિર્ધારિત રીતે આ પ્રકારના ચલમાં અનિશ્ચિત કિંમત (garbage value) નો ઉપયોગ કરવામાં આવે છે.

સ્ટેટિક ચલ (Static Variable)

જ્યારે ચલનો 'સ્ટેટિક' સંગ્રાહક વર્ગ સાથે ઉપયોગ કરવામાં આવે છે ત્યારે નિશ્ચિત વિસ્તાર માટે તે ચલની કિંમત કાયમી બનાવી શકાય છે. સ્ટેટિક સંગ્રાહક વર્ગ ધરાવતા ચલની કિંમતને પ્રાઇમરી મેમરીમાં કાયમી ધોરણે સાચવવામાં આવે છે તથા પૂર્વનિર્ધારિત રીતે તેમાં શૂન્ય કિંમત મૂકવામાં આવે છે. ચલનો સ્ટેટિક સંગ્રાહક વર્ગ વ્યાખ્યાયિત કરવા માટે નીચે આપેલ વાક્યરચનાનો ઉપયોગ કરવામાં આવે છે :

static datatype identifier;

ઉદાહરણ રૂપે, static int max; વ્યાખ્યા દર્શાવે છે કે max ચલ પૂર્ણાંક પ્રકારનો છે અને તેનો સંગ્રાહક વર્ગ સ્ટેટિક છે.

સારાંશ

આ પ્રકરણમાં આપણે મૂળભૂત ડેટા પ્રકારો int, float, char અને void વિશે અભ્યાસ કર્યો. typedef, enum અને array જેવા ઉપયોગકર્તા દ્વારા નિર્મિત અને તારવેલા ડેટા પ્રકારો વિશે પણ માહિતી મેળવી. નિરૂપક પ્રક્રિયક દ્વારા ચલમાં કેવી રીતે કિંમત ઉમેરવી તે પણ આપણે જોયું. ત્યાર પછી આપણે ગાણિતિક, સંબંધસૂચક, વધારા-ઘટાડા સૂચક, શરતી, તાર્કિક, બિટવાઇઝ અને વિશિષ્ટ પ્રક્રિયકો વિશે અભ્યાસ કર્યો. જુદા જુદા પ્રક્રિયકોનો ઉપયોગ કરી પદાવલિની રચના કરતાં તથા તેનું મૂલ્યાંકન કરતાં પણ આપણે શીખ્યા. અંતમાં સી ભાષા દ્વારા પૂરા પાડવામાં આવતા ચાર સંગ્રાહક વર્ગો વિશે માહિતી મેળવી.

શિક્ષકોને સૂચના (Instruction for Teachers)

અહીં ચર્ચવામાં આવેલા તમામ ડેટા પ્રકારો ANSI C99 ધારાધોરણ આધારિત છે. અહીં આપવામાં આવેલાં ઉદાહરણ પ્રક્રિયકોને સમજવા માટે છે. શિક્ષક વધુ સ્વાધ્યાય બનાવી વિદ્યાર્થીઓને આપી શકે છે.

સ્વાધ્યાય

1. ડેટા પ્રકાર એટલે શું ? તેનું મહત્ત્વ જણાવો.

2. void ડેટા પ્રકારનું મહત્ત્વ જણાવો.

3. ઉપયોગકર્તા દ્વારા વ્યાખ્યાયિત ડેટા પ્રકારનું મહત્ત્વ જણાવો.

4. તારવેલા ડેટા પ્રકારનું મહત્ત્વ જણાવો.

5. નીચેના વિધાનો ખરાં છે કે ખોટાં તે જણાવો :

(a) char ડેટા પ્રકાર માટે બે બાઇટ જગ્યા જરૂરી છે.

(b) ચિહ્નરહિત (unsigned) પૂર્ણાંક ચલ શૂન્યથી ઓછી સંખ્યાનો સંગ્રહ કરી શકે છે.

(c) double ડેટા પ્રકારનો ઉપયોગ પૂર્ણાંક સંખ્યાઓનો સંગ્રહ કરવા માટે છે.

(d) int ડેટા પ્રકારનો વિસ્તાર વધારવા માટે તેની આગળ long કી-વર્ડ ઉમેરવામાં આવે છે.

(e) અપૂર્ણાંક સંખ્યાઓનું વૈજ્ઞાનિક સ્વરૂપ exponent * 10 mantissa તરીકે રજૂ કરવામાં આવે છે.

6. આપેલા વિકલ્પોમાંથી યોગ્ય વિકલ્પ પસંદ કરો :

(1) પૂર્ણાંક ડેટા પ્રકાર માટે નીચેનામાંથી કયા કી-વર્ડનો ઉપયોગ કરવામાં આવે છે ?

(a) integer (b) Integer (c) INTEGER (d) int

(2) નીચેનામાંથી કયો વિકલ્પ ખાલી કિંમતનો નિર્દેશ કરે છે ?

(a) void (b) Void (c) char (d) float

(3) gcc કંપાઇલર દ્વારા float પ્રકારને કયા કદની મેમરી આપવામાં આવે છે ?

(a) 1 (b) 2 (c) 4 (d) 6

(4) સી ભાષામાં એક અક્ષર વ્યાખ્યાયિત કરવા માટે નીચેનામાંથી કયો કી-વર્ડ યોગ્ય છે ?

(a) char (b) character (c) CHAR (d) CHARACTER

(5) સી ભાષામાં નીચેનામાંથી કયો ડેટા પ્રકાર ઉપયોગકર્તા દ્વારા નિર્મિત છે ?

(a) int (b) enum (c) char (d) float

(6) સી ભાષામાં આવેલા ડેટા પ્રકારને ઉપનામ આપવા માટે નીચેનામાંથી કયા કી-વર્ડનો ઉપયોગ કરવામાં આવે છે ?

(a) enum (b) def (c) pointer (d) typedef

(7) સી ભાષામાં "&" નિશાની કયા પ્રકારનો પ્રક્રિયક છે ?

(a) સંબંધસૂચક (b) ગાણિતિક (c) તાર્કિક (d) બિટવાઇઝ

(8) સી ભાષામાં "!" નિશાની કયા પ્રકારનો પ્રક્રિયક છે ?

(a) સંબંધસૂચક (b) ગાણિતિક (c) તાર્કિક (d) બિટવાઇઝ

(9) સી ભાષામાં "=" નિશાનીનો ઉપયોગ કયા પ્રક્રિયક તરીકે કરવામાં આવે છે ?

(a) સમાનતા (b) નિરૂપણ (c) નોંધ (d) સમીકરણ

(10) સી ભાષામાં value++ નો અર્થ શું કરી શકાય?

(a) પોસ્ટ ઇન્ક્રિમેન્ટ (b) પોસ્ટ ડિક્રિમેન્ટ (c) પ્રી-ઇન્ક્રિમેન્ટ (d) પ્રી ડિક્રિમેન્ટ

પ્રાયોગિક સ્વાધ્યાય

1. આપેલ મીટરને મિલીમીટરમાં ફેરવવા માટેનો સી પ્રોગ્રામ લખો.

2. c = (f–32) * 5/9 સમીકરણનો ઉપયોગ કરી આપેલ ફેરનહીટને સેલ્સિયસમાં ફેરવવા માટેનો સી પ્રોગ્રામ લખો.

3. ત્રણ સંખ્યાઓની સરેરાશ શોધવા માટેનો સી પ્રોગ્રામ લખો.

4. 2000 ચોમી ના વિસ્તારમાં 50 ચો સેમીની એક એવી કુલ કેટલી લાદી જડી શકાય તે શોધવા માટેનો પ્રોગ્રામ લખો.

5. લંબચોરસનું ક્ષેત્રફળ શોધવા માટેનો સી પ્રોગ્રામ લખો.

6. આપેલ સંખ્યા અન્ય આપેલ સંખ્યા સાથે નિઃશેષ ભાજ્ય છે કે નહીં તે શોધવા માટેનો સી પ્રોગ્રામ લખો.

7. ₹ P ની લોન R% વ્યાજના દર સાથે N વર્ષો માટે લેવામાં આવી હોય તો તે પરથી ચક્રવૃદ્ધિ વ્યાજ શોધવા માટેનો સી પ્રોગ્રામ લખો. P, R અને Nની કિંમતો વિદ્યાર્થી દ્વારા આપવામાં આવશે.

8. ઘનનું ઘનફળ શોધવા માટેનો સી પ્રોગ્રામ લખો.

9. વર્તુળનું ક્ષેત્રફળ શોધવા માટેનો સી પ્રોગ્રામ લખો.

10. ₹ P ની લોન R% વ્યાજના દર સાથે N વર્ષો માટે લેવામાં આવી હોય તો તે પરથી સાદું વ્યાજ શોધવા માટેનો સી પ્રોગ્રામ લખો. P, R અને Nની કિંમતો વિદ્યાર્થી દ્વારા આપવામાં આવશે.