باب 5

فنکشنز

تدریسی مقاصد: (Student Learning OutComes)

- فنکشنز کا تصور اور ان کی اقسام کی وضاحت
- فنکشنز کے استعمال کے فوائد کی وضاحت
- فنکشنز کے سگنیچر (نام، آرگیومینٹس، ریٹرن ٹائپ) کی وضاحت
- فنکشنز سے متعلق اصطلاحات کی وضاحت۔
    - فنکشن کی تعریف
    - فنکشن کا استعمال
- C لینگویج کو استعمال کرتے ہوئے درج ذیل فنکشنز لکھنا:
    - ایک فنکشن جو دو انٹیجر (integer) ویری ایبلز بطور آرگیومینٹ لے اور ان کی حاصل واپس کرے۔
    - ایک فنکشن جو تین متغیرات لے اور ان کا درمیان والا نمبر واپس کرے۔

یونٹ کا تعارف (Unit Introduction)

کسی بھی مسئلے کو حل کرنے کی اچھی حکمتِ عملی یہ ہے کہ اسے چھوٹے چھوٹے حصوں میں تقسیم کر دیا جائے۔ ایک ایک حصے کا حل نکال کر ان سب کو اکٹھا کرنے سے پورے مسئلے کا حل مل جاتا ہے۔ سارا وقت پورے مسئلے کے بارے میں سوچنے کے بجائے ایک وقت میں ایک چھوٹے حصے پر غور کرنا آسان ہوتا ہے۔ مسئلے کا حل نکالنے کی اس حکمتِ عملی کو تقسیم کرنا اور فتح کرنا (Divide and conquer) کہتے ہیں۔ C پروگرامنگ لینگوئج میں ہمارے پاس فنکشنز ہوتے ہیں جو پروگرامنگ کے سوال کو حل کرنے کے لیے تقسیم کرنے اور فتح کرنے کی حکمتِ عملی استعمال کرتے ہیں۔ اس باب میں ہم فنکشنز کا تصور ان کے فوائد اور ان کے ساتھ کام کرنے کا طریقہ سیکھیں گے۔

5.1 فنکشنز (Functions)

فنکشن سٹیٹمنٹس کا ایک بلاک ہے جو ایک خاص کام انجام دیتا ہے مثلاً printf ایک فنکشن ہے جو کمپیوٹر کی سکرین پر کچھ بھی دکھانے کے لیے استعمال ہوتا ہے۔ scanf ایک فنکشن ہے جو صارف سے ان پٹ لینے کے لیے استعمال ہوتا ہے۔ ہر پروگرام میں ایک main فنکشن ہوتا ہے۔ جو صارف کے مطلوب افعال پروگرام کو سرانجام دیتا ہے۔ اسی طرح ہم اور فنکشنز بھی لکھ سکتے ہیں۔ اور انھیں کئی مرتبہ استعمال کر سکتے ہیں۔

5.1.1 فنکشنز کی اقسام

بنیادی طور پر فنکشنز کی دو اقسام ہیں:

1۔ بِلٹ اِن فنکشنز (Built-in Function)

2۔ یوزر ڈیفائنڈ فنکشنز (User-Defined Function)

بلٹ ان فنکشنز (Built in Function)

وہ فنکشنز جو C کی سٹینڈرڈ لائبریری میں موجود ہیں بلٹ ان فنکشنز کہلاتے ہیں۔ یہ فنکشنز عام طور پر ریاضی کے حساب کتاب، سٹرنگ آپریشنز، اِن پٹ اور آؤٹ پٹ آپریشنز وغیرہ انجام دیتے ہیں مثلاً printf اور scanf بلٹ ان فنکشنز ہیں۔

یوزر ڈیفائنڈ فنکشنز (User-defined Function)

وہ فنکشنز جو پروگرامر ڈیفائن کرتا ہے یوزر ڈیفائنڈ فنکشنز کہلاتے ہیں اس باب میں ہم یوزر ڈیفائنڈ فنکشنز لکھنا سیکھیں گے۔

5.1.2 فنکشنز کے فوائد

فنکشنز سے ہمیں درج ذیل فوائد حاصل ہوتے ہیں:

1۔ دوبارہ استعمال (Reusability)

فنکشنز کے ذریعے ہم کوڈ کو دوبارہ استعمال کر سکتے ہیں ۔ اس کا مطلب یہ ہے کہ جب بھی ہمیں ایک فنکشن کی فنکشنیلیٹی (Functionality) کی ضرورت ہو ہم اس فنکشن کو کال (call) کر سکتے ہیں۔ ہمیں سٹیٹمنٹس کا ایک سیٹ بار بار نہیں لکھنا پڑتا۔

2۔ کاموں کو الگ کرنا (Separation of Tasks)

فنکشن کے ذریعے ہم ایک کام کرنے کے کوڈ کو دوسرے کام کے کوڈ سے الگ کر سکتے ہیں ۔ اگر ہمیں ایک فنکشن میں مسئلہ ہو تو اسے حل کرنے کے لیے پورے پروگرام کو چیک نہیں کرنا پڑتا۔ ہمیں صرف ایک فنکشن پر غور کرنا ہوتا ہے۔

3۔ مسئلے کی پیچیدگی سے نمٹنا (Handling the Complexity of Problem)

اگر ہم پورا پروگرام ایک پروسیجر کے طور پر لکھیں تو اس کی دیکھ بھال (manage) کرنا مشکل ہو جاتا ہے۔ فنکشنز کے ذریعے ہم پروگرام کو چھوٹے حصوں میں تقسیم کرتے ہیں اس سے مسئلے کی پیچیدگی کم ہو جاتی ہے۔

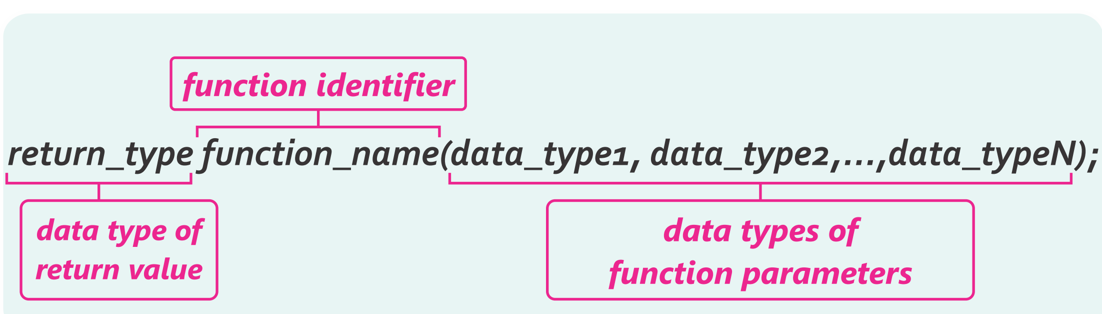
4۔ پڑھنے کی صلاحیت (Readability)

پروگرام کو کئی فنکشنز میں تقسیم کرنے سے اس کو پڑھنا (سمجھنا) زیادہ آسان ہو جاتا ہے۔

5.1.3 فنکشن کا سگنیچر (Signature of Function):

فنکشن سٹیٹمنٹ کا ایک بلاک ہوتا ہے جو کچھ اِن پٹ لیتا ہے۔ فنکشن کی اِن پُٹس کو پیرامیٹرز کہتے ہیں اور آؤٹ پٹ کو ریٹرن ویلیو (Return Value) کہتے ہیں۔ ایک فنکشن کے ایک سے زیادہ پیرامیٹرز تو ہو سکتے ہیں لیکن وہ ایک ہی قیمت ریٹرن کر سکتا ہے زیادہ نہیں۔

فنکشن سگنیچر فنکشن کی اِن پُٹس اور آؤٹ پٹ ڈیفائن کرتا ہے۔ فنکشن کے سگنیچر کا عام ڈھانچہ یہ ہے۔



فنکشن سگنیچرز کی مثالیں:

ٹیبل 5.1 میں کچھ فنکشنز اور ان کے سگنیچرز کی تفصیل دی گئی ہے:

فنکشن کی تفصیل	فنکشن سگنیچر
ایک فنکشن جو ایک انٹیجر (int) ان پُٹ لیتا ہے۔ اور اُس کا مربع واپس کرتا ہے۔	`int square (int);`
ایک فنکشن جو مستطیل کی لمبائی اور چوڑائی اِن پُٹ لے اور اس کا احاطہ ریٹرن کرے۔	`float perimeter (float, float);`
ایک فنکشن جو تین انٹیجر (integers) اِن پُٹ لے اور ان میں سے سب سے بڑی ویلیو ریٹرن کرلے۔	`int largest (int, int, int);`
ایک فنکشن جو دائرے کا رداس اِن پُٹ لے اور رقبہ ریٹرن کرے۔	`float area (float);`
ایک فنکشن جو ایک کریکٹر اِن پُٹ لے اور اگر وہ حرفِ علت ہو تو 1 ریٹرن کرے ورنہ 0 ریٹرن کرے۔	`int isVowel (char);`

ٹیبل 5.1 کچھ فنکشنز اور ان کے سگنیچرز

5.1.4 فنکشن کو ڈیفائن کرنا (Defining a function)

ایک فنکشن کے سگنیچر سے یہ نہیں پتا چلتا کہ وہ کس کام کے لیے ہے۔ اس کے لیے فنکشن ڈیفینیشن (Function Definition) ہوتی ہے۔ ایک فنکشن ڈیفینیشن کا ڈھانچہ کچھ اس طرح سے ہے:

```
return_type function_name (data_type var1, data_type var2,.., data_type varN)
{
        Body of the function
}
```

فنکشن کی باڈی ان سٹیٹمنٹس کا سیٹ ہوتا ہے جنھیں چلا کر فنکشن کوئی خاص کام سرانجام دیتا ہے۔ فنکشن سگنیچر کے فوراً بعد { } قوسین میں لکھی گئی سٹیٹمنٹس فنکشن کی باڈی بناتی ہیں۔ درج ذیل مثال ایک فنکشن کو ڈیفائن کرتی ہے showPangram() جو نہ تو کچھ اِن پُٹ لیتا ہے نہ کچھ ریٹرن کرتا ہے بس کمپیوٹر کی سکرین پر "A Quick Brown Fox Jumps Over the Lazy Dog" دکھاتا ہے۔

EXAMPLE CODE 5.1

```
void showPangram()        ← function name
{
      printf("\nA quick brown fox jumps over the lazy
      dog.\n");
}
```

چوں کہ درج بالا فنکشن کچھ ریٹرن نہیں کرتا اس لیے اس کی ریٹرن ٹائپ void ہے۔

آیئے! اب ایک اور مثال لیتے ہیں جو دو انٹیجر (integer) اِن پُٹ لے اور ان کی جمع ریٹرن کرے۔

EXAMPLE CODE 5.2

```
int add(int x, int y)     ← parameters of function / function name / return type
{
      int result;
      result = x + y;
      return result;
}
```

فنکشن کے اندر return ایک مطلوبہ لفظ ہے جو کالنگ (calling) فنکشن کو قیمت ریٹرن کرنے کے لیے استعمال ہوتا ہے۔

اہم نوٹ:

ایک فنکشن ایک سے زیادہ قیمتیں ریٹرن نہیں کرسکتا مثلاً:

درج ذیل سٹیٹمنٹ کے نتیجے میں کمپائلر ایرر دیتا ہے۔

return (4,5);

اہم نوٹ:

ایک فنکشن میں ایک سے زیادہ ریٹرن سٹیٹمنٹس ہو سکتی ہیں لیکن جیسے ہی ایک ریٹرن سٹیٹمنٹ چلتی ہے فنکشن کی کال ختم ہو جاتی ہے فنکشن کی باڈی کی باقی سٹیٹمنٹس نہیں چلتیں۔

فنکشن کا استعمال:

ہمیں ایک فنکشن کو کال کرنا پڑتا ہے تا کہ وہ پروگرام کو سونپا گیا کام انجام دے فنکشن کال کے لیے یہ ڈھانچہ استعمال ہوتا ہے۔

```
function_name(value1, value2,…, valueN);
```

مثال کے طور پر درج ذیل پروگرام کو دیکھیں۔

EXAMPLE CODE 5.3

```
void main()
{
      printf("Hello from main()");
      showPangram();  <------ function call
      printf("Welcome back to main()");
}
```

Output:

```
Hello from main()
A quick brown fox jumps over the lazy dog.
Welcome back to main()
```

ہم دیکھ سکتے ہیں کہ ایک پروگرام main فنکشن سے چلنا شروع کرتا ہے۔ جب کوئی فنکشن کال (مستطیل کے اندر) آتی ہے تو کنٹرول اس فنکشن کو ٹرانسفر ہو جاتا ہے۔ کال کیے گئے فنکشن کے ایگزیکیوٹ ہونے کے بعد کنٹرول واپس اس فنکشن کے پاس چلا جاتا ہے جس نے فنکشن کال کیا ہوتا ہے جیسے اوپر والی مثال میں main()۔

درج ذیل پروگرام 2 نمبر ان پٹ کے طور پر لیتا ہے اور اُن کا مجموعہ ریٹرن کرتا ہے۔

درج ذیل کوڈ میں مستطیل کے اندر جو سٹیٹمنٹ ہے وہ پچھلے سیکشن میں ڈیفائن کیے ہوئے فنکشن ''add'' کو کال کرتی ہے۔

EXAMPLE CODE 5.4

```
void main ()
{
      int n1, n2, sum;
      scanf ("%d%d", &n1, &n2);
      sum = add (n1, n2);
      printf ("Sum is %d", sum);
}
```

function name
function call
function arguments

- فنکشن کال میں n1 اور n2 فنکشن ()add کے آرگومینٹس ہیں۔
- فنکشن ()add سے ریٹرن ہونے والے نتیجے کو محفوظ کرنے کے لیے متغیر sum ڈکلیئر کیا گیا ہے۔
- فنکشن کو بطور آرگومینٹ پاس کیے گئے ویری ایبلز میں کوئی تبدیلی نہیں آتی۔ فنکشن ان ویری ایبلز کی ایک کاپی بناتا ہے اور صرف اس کاپی میں تبدیلیاں کرتے ہیں۔
- درج بالا مثال میں جب n1 اور n2 پاس کیے جاتے ہیں تو فنکشن ان ویری ایبلز کی کاپیاں بنالیتا ہے۔ ویری ایبل n1 کی کاپی x ہے اور ویری ایبل n2 کی کاپی y ہے۔

اہم نوٹ:

جو قیمتیں فنکشن کو پاس کی جاتی ہیں وہ آرگومنٹس کہلاتی ہیں جب کہ فنکشن ڈیفینیشن میں جن ویری ایبلز میں یہ قیمتیں جاتی ہیں وہ فنکشن کے پیرامیٹرز کہلاتے ہیں۔

اُوپر دی گئی مثال میں ویری ایبلز n1 اور n2 آرگومنٹس ہیں جو فنکشن ()add کو پاس کیے گئے ہیں جبکہ فنکشن ()add کے اندر ویری ایبلز x اور y اس کے پیرامیٹرز ہیں

اہم نوٹ:

ضروری نہیں کہ فنکشن کو جو ویری ایبلز پاس کیے جائیں ان کے نام وہی ہوں جو فنکشن کے پیرامیٹرز کے نام ہوں۔ البتہ ہم ایک جیسے نام بھی استعمال کر سکتے ہیں یہاں ایک اہم نکتہ یہ ہے کہ اگر ہم ایک جیسے نام استعمال کریں گے تو بھی فنکشن میں استعمال ہونے والے ویری ایبلز اصل ویری ایبلز کی کاپی ہوں گے۔ یہ درج ذیل مثال سے واضح کیا گیا ہے۔

EXAMPLE CODE 5.5 <\>

```
#include<stdio.h>
void fun(int x, int y)
{
      x = 20;
      y = 10;
      printf("Values of x and y in fun(): %d %d", x, y);
}
void main()
{
      int x = 10, y = 20;
      fun(x, y);
      printf("Values of x and y in main(): %d %d", x, y);
}
```

Output:

```
Values of x and y in fun(): 20 10
Values of x and y in main(): 10 20
```

اہم نوٹ:

پروگرام میں فنکشنز کو ترتیب دیتے ہوئے درج ذیل نکات ذہن میں رکھیں۔

1۔اگر کال کیے گئے فنکش کی ڈیفینیشن کال کرنے والے فنکشن کی ڈیفینیشن سے پہلے آئے تو فنکشن سگنیچر کال کرنے والے فنکشن کی ڈیفینیشن سے پہلے لکھنا ضروری ہے۔

2۔اگر کال کیے گئے فنکشن کی ڈیفینیشن کال کرنے والے فنکشن کے بعد میں آئے تو کال کیے گئے فنکشن کا سگنیچر اِس کال کرنے والے فنکشن سے پہلے لکھنا ضروری ہے۔

نیچے دیے گئے دونوں کوڈ سٹرکچرز درست ہیں۔

a) `int add(int, int);`
`void main()`
`{`
`    printf(”%d ”add(4, 5));`
`}`
`int add(int a, int b)`
`{`
`    return a + b;`
`}`
	b) `int add(int a, int b)`
	`{`
	`    return a + b;`
	`}`
	`void main()`
	`{`
	`    printf(”%d ”add(4, 5)`
	`}`

5.1 پروگرامنگ ٹائم (Programming Time)

ایک فنکشن prime() لکھیں جو ایک نمبر اِن پٹ لے اور 1 ریٹرن کرے اگر نمبر مفرد ہو تو نہیں تو 0 ریٹرن کرے۔اس فنکشن کو main() میں استعمال کریں۔

Program:

```
#include <stdio.h>
int prime (int n)
{
      for (int i = 2; i < n; i++)
            if(n % i == 0)
                  return 0;
      return 1;
}
```

جاری ہے۔

```
void main()
{
        int x;
        printf ("Please enter a number: ");
        scanf ("%d", &x);
        if(prime(x))
                printf ("%d is a Prime Number",x);
        else
                printf ("%d is not a Prime Number",x);
}
```

5.2 پروگرامنگ ٹائم (Programming Time)

پرابلم:

ایک فنکشن لکھیں جو ایک مثبت نمبر ان پٹ کے طور پر لے اور 0 سے لے کر اُس نمبر تک نمبروں کو جمع کرے اور حاصل جمع ریٹرن کرے۔

Program:

```
int digitsSum(int n)
{
        int sum = 0;
        for(int i = 0; i <= n; i++)
        {
                sum = sum + i;
        }
        return sum;
}
void main()
{
        int number;
        printf("Please enter a positive number: ");
        scanf("%d", &number);
```

جاری ہے۔

```
      if(number >= 0)
      {
            int sum = digitsSum(number);
            printf("The  sum  of numbers upto  given  number  is
%d",sum);
      }
      else
            printf("You entered a negative number.");
}
```

خلاصہ

- فنکشن سٹیٹمنٹس کا ایک بلاک ہے جو ایک خاص کام انجام دیتا ہے۔
- C-سٹینڈرڈ لائبریری میں موجود فنکشنز بلٹ ان فنکشنز کہلاتے ہیں۔
- پروگرامر جو فنکشنز ڈیفائن کرتا ہے وہ یوزر ڈیفائنڈ فنکشنز کہلاتے ہیں۔
- فنکشن استعمال کرنے کے کچھ فوائد یہ ہیں: کوڈ کو دوبارہ استعمال کرنا، کاموں کو الگ کرنا، مسئلے کی پیچیدگی کو کم کرنا اور کوڈ کو پڑھنے کے قابل بنانا۔
- فنکشن سیگنیچر فنکش کے نام، ان پٹس اور آؤٹ پٹ کی وضاحت کرتا ہے۔
- فنکشن کو اس طرح ڈیفائن کیا جاسکتا ہے۔

```
return_type name (Parameters)
{
   Body of the Function
}
```

- فنکشن کی ریٹرن ٹائپ اس قیمت کی ڈیٹا ٹائپ ہوتی ہے جو فنکشن ریٹرن کرتا ہے۔
- فنکشن کا نام اس کے کام سے متعلق ہونا چاہیے۔
- پیرامیٹرز مختلف ڈیٹا ٹائپس کے متغیرات ہوتے ہیں جن میں فنکشن کو بطور ان پٹ پاس کی گئی قیمتیں رکھی جاتی ہیں۔
- فنکشن کی باڈی ان سٹیٹمنٹس کا سیٹ ہوتی ہے جنھیں چلا کر فنکشن مخصوص کام سرانجام دیتا ہے۔
- ایک فنکشن کو کال کرنے سے مراد اس فنکشن کو کنٹرول ٹرانسفر کرنا ہے۔
- فنکشن کال کے دوران جو قیمتیں فنکشن کو پاس کی جاتی ہیں وہ آرگیومینٹس کہلاتی ہیں۔
- جیسے ہم main() سے باقی فنکشنز کو کال کر سکتے ہیں ایسے ہی ایک یوزر ڈیفائنڈ فنکشن سے دوسرے یوزر ڈیفائنڈ فنکشن کو بھی کال کر سکتے ہیں۔

مشق

سوال نمبر 1: کثیر الانتخابی سوالات۔

1) فنکشن بلٹ ان یا ____________ ہو سکتے ہیں:

الف) ایڈمن ڈیفائنڈ ب) سرور ڈیفائنڈ ج) یوزر ڈیفائنڈ د) دونوں الف اور ج

2) C-سٹینڈرڈ لائبریری میں موجود فنکشنز ____________________ کہلاتے ہیں:

الف) یوزر ڈیفائنڈ ب) بلٹ ان ج) تکرار پر مبنی د) تکراری

3) فنکشن کو پاس کی گئی قیمتیں ____________ کہلاتی ہیں:

الف) باڈیز ب) ریٹرن ٹائپس ج) ارے د) آرگیومنٹس

4) char cd() {return = 'a';} اس فنکشن میں "char" ______________ ہے:

الف) باڈی ب) ریٹرن ٹائپ ج) ارے د) آرگیومینٹس

5) فنکشنز کو استعمال کرنے کے فوائد ________________ ہیں۔:

الف) پڑھے جانے کی صلاحیت ب) بار بار استعمال ج) ڈیبگنگ میں آسانی د) پہلے تینوں

6) اگر فنکشن باڈی میں تین ریٹرن سٹیٹمنٹس ہوں تو ان میں سے ____________________ چلیں گی:

الف) ایک ب) دو ج) تین د) پہلی اور آخری

7) پڑھے جانے کی صلاحیت (readability) کوڈ کو ________________ کرنے میں مدد دیتی ہے:

الف) سمجھنے ب) تبدیل کرنے ج) ڈیبگ کرنے د) پہلے تینوں

8) ________________ سے مراد کوڈ ایک اور فنکشن میں ٹرانسفر کرنا ہے:

الف) کالنگ ب) ڈیفائننگ ج) ری۔رائٹنگ د) انکلیوڈنگ (including)

سوال نمبر 2: درج ذیل کی تعریف کریں۔

1) فنکشنز 2) بلٹ۔ان فنکشنز 3) فنکشن پیرامیٹرز 4) بار بار استعمال (reusability)

5) فنکشن کو کال کرنا

سوال نمبر 3: درج ذیل سوالات کے مختصر جوابات لکھیں۔

1) آرگیومینٹس اور پیرامیٹرز میں کیا فرق ہے؟ ایک مثال دیں۔

2) فنکشن ڈیفینیشن کے حصوں کی فہرست لکھیں۔

3) کیا یہ ضروری ہے کہ فنکشن ڈیفینیشن اور فنکشن کال کی ڈیٹا ٹائپس میں ہم آہنگی ہو؟ مثال کے ساتھ جواب کی توثیق کریں۔

4) فنکشنز استعمال کرنے کے فوائد کی وضاحت کریں۔

5) آپ کی۔ورڈ return کے بارے میں کیا جانتے ہیں؟

سوال نمبر 4: کوڈ کے درج ذیل حصوں میں ایررز تلاش کریں۔

a)
```
void sum (int a, int b)
{
        return a + b;
}
```

b)
```
void message ();
{
        printf (“Hope you are fine :)”);
        return 23;
}
```

c)
```
int max (int a; int b)
{
        if (a > b)
                return a;
        return b;
}
```

d)
```
int product (int n1, int n2)
        return n1*n2;
```

e)
```
int totalDigits (int x)
{
        int count = 0;
        for (int i = x; i >= 1, i = i/10)
                count++;
        return count
};
```

سوال نمبر 5: کوڈ کے درج ذیل حصوں کی آؤٹ پٹ تحریر کریں۔

a)
```
int xyz (int n)
{
    return n + n;
}
int main()
{
    int p = xyx(5);
    p = xyz(p);
    printf (“%d “,p);
}
```

b)
```
void abc (int a, int b, int c)
{
    int sum = a + b + c;
}
int main()
{
    int x = 4, y = 7, z = 23, sum1 = 0;
    abc (x, y, z);
    printf (“%d %d %d” x, y ,z);
}
```

c)
```
int aa (int x)
{
    int p = x / 10;
    x++;
    p = p + (p * x);
    return p;
}
int main()
{
    printf (“We got %d “, aa(aa(23)));
}
```

d)
```
float f3(int n1, int n2)
{
      n1 = n1 + n2;
      n2 = n2 - n1;
      return 0;
}
int main()
{
      printf (“%f\n”, f3(3, 2));
      printf (“%f\n”, f3(10, 6));
}
```

پروگرامنگ کی مشقیں

مشق نمبر1:

ایک انٹیجر (integer) x کا مربع معلوم کرنے کے لیے ;int square (int x) فنکشن لکھیں۔

مشق نمبر2:

ایک فنکشن ;int power(int x, int y) لکھیں جو x^y معلوم کرکے ریٹرن کرے۔

مشق نمبر3:

ایک نمبر کا فیکٹوریل نکالنے کا فنکشن لکھیں۔

مشق نمبر4:

ایک فنکشن لکھیں جو مثلث کے تین زاویے لے اور بتائے کہ یہ زاویے صحیح مثلث کے ہیں یا نہیں۔ ایک صحیح مثلث وہ ہوتی ہے جس کے تینوں زاویوں کا مجموعہ 180 ہو۔

مشق نمبر5:

ایک فنکشن لکھیں جو رقم اور انٹرسٹ ریٹ لے اور انٹرسٹ کی رقم ریٹرن کرے۔

مشق نمبر6:

ایک فنکشن لکھیں جو ایک نمبر ان پٹ لے اور سپیسز کے ساتھ اس کے ہندسے پرنٹ کرے۔

مشق نمبر7:

کسی نمبر کا ٹیبل پرنٹ کرنے کا فنکشن لکھیں۔

اصطلاحات

n\: یہ بتاتا ہے کہ کرسر کو اگلی لائن کے شروع میں لے کر جانا ہے۔

t\: یہ بتاتا ہے کہ کرسر کو افقی طور پر اگلے ٹیب سٹاپ پر لے کر جانا ہے۔ایک ٹیب سٹاپ آٹھ سپیسز کا مجموعہ ہوتا ہے۔

آرگیومینٹس: وہ قیمتیں جو فنکشن کال کے دوران فنکشن کو پاس کی جاتی ہیں۔

ارتھمیٹک اوپریٹرز: یہ ارتھمیٹک فنکشنز کی قیمت نکالنے کے لیے ڈیٹا پر حساب کتاب کرنے میں استعمال ہوتے ہیں۔% ,/ ,*,- ,+ ارتھمیٹک اوپریٹرز ہیں۔

ارے کی انیشلائزیشن: پہلی مرتبہ ارے میں قیمتیں لکھنا۔ایک ارے کو ڈکلیئریشن کے وقت یا اس کے بعد انیشلائز کیا جاسکتا ہے۔

ارے کا سائز: زیادہ سے زیادہ ایلیمنٹس کی تعداد جو ایک ارے میں رکھے جاسکتے ہیں۔

ارے: ایک ڈیٹا سٹرکچر جو کمپیوٹر کی میموری میں اکٹھی لوکیشنز پر ایک ہی ڈیٹا ٹائپ کی ایک سے زیادہ قیمتیں رکھ سکتا ہے۔

اسائنمنٹ اوپریٹر: یہ ایک متغیر میں قیمت رکھنے یا ایک متغیر کو دوسرے متغیر کی قیمت منسوب کرنے کے لیے استعمال ہوتا ہے۔

ایڈیٹر: ایک سافٹ ویئر جس کے ذریعے پروگرامر کمپیوٹر پروگرام لکھ سکتا ہے اور اس میں ترمیم کرسکتا ہے۔

اسکیپ سیکوئنس: یہ printf کو بتاتا ہے کہ عام طریقہ کار سے ہٹ کر کام کرنا ہے۔
یہ اسکیپ کریکٹر (\) اور ایسے کریکٹر کا مجموعہ ہوتا ہے جس سے خاص فنکشنیلٹی منسوب ہو۔

انڈیکس: یہ ارے کے ایلیمنٹس تک رسائی حاصل کرنے کے لیے استعمال ہوتا ہے۔متغیرات کو بھی ارے انڈیکسز کے طور پر استعمال کیا جاسکتا ہے۔

انٹیجر ڈیٹا ٹائپ: ایک ڈیٹا ٹائپ جس میں انٹیجر کانسٹینٹس محفوظ کیے جاتے ہیں۔اس کا سائز 4 بائٹس ہے۔

انٹیگریٹڈ ڈیویلپمنٹ انوائرمنٹ: ایک ایسا سافٹ ویئر جو پروگرامر کو کمپیوٹر پروگرام لکھنے اور چلانے کے لیے پروگرامنگ انوائرمنٹ فراہم کرے۔

if سٹیٹمنٹ: یہ کنڈیشن سے منسوب کوڈ تب چلاتی ہے جب کنڈیشن پوری ہو ورنہ نہیں چلاتی۔

if-else سٹیٹمنٹ: یہ if سٹیٹمنٹ سے منسوب سٹیٹمنٹس کا سیٹ چلاتی ہے۔اگر کنڈیشن پوری ہو ورنہ else سٹیٹمنٹ سے منسوب سٹیٹمنٹس کا سیٹ چلاتی ہے۔

بنیادی اوپریٹرز: ان میں ارتھمیٹک اوپریٹرز، اسائنمنٹ اوپریٹر، ریلیشنل اوپریٹر اور منطقی اوپریٹر شامل ہیں۔

بائنری اوپریٹرز: انھیں دو اوپرینڈز درکار ہوتے ہیں۔

بولین: ایک ڈیٹا ٹائپ جس میں true یا false رکھا جاسکتا ہے۔

بلٹ ان فنکشنز: وہ فنکشنز جو C کی سٹینڈرڈ لائبریری میں موجود ہوں۔

پیرامیٹرز: مختلف ڈیٹا ٹائپس کے متغیرات جن میں فنکشن کو پاس کی گئی قیمتیں رکھی جاتی ہیں۔

پروگرامر: وہ شخص جسے معلوم ہو کہ ایک صحیح کمپیوٹر پروگرام کیسے لکھا جاتا ہے۔

پروگرامنگ انوائرمنٹ: پروگرامنگ کے تمام اہم آلات کا مجموعہ۔

پروگرامنگ لینگویجز: وہ خاص زبانیں جن میں/ پروگرام لکھتے ہیں۔

اصطلاحات

ترجیح: اس سے معلوم ہوتا ہے کہ کونسا اوپریشن پہلے انجام دینا ہے۔

ٹرنری اوپریٹرز: انھیں تین اوپرینڈ درکار ہوتے ہیں۔

ڈیٹا سٹرکچر: ایک کنٹینر جس میں ایک خاص ترتیب سے ڈیٹا آئٹمز کے مجموعے کو محفوظ کیا جاتا ہے۔

ڈیٹا ٹائپ: یہ بتاتی ہے کہ متغیر میں کس قسم کی قیمت محفوظ کی جا سکتی ہے۔

ریلیشنل اوپریٹرز: یہ دو قیمتوں میں موازنہ کر کے ان کا تعلق بتاتے ہیں۔

ریٹرن ٹائپ: یہ اس قیمت کی ڈیٹا ٹائپ ہے جو فنکشن ریٹرن کرتا ہے۔

سیکوئنشل کنٹرول: تمام سٹیٹمنٹس دی گئی ترتیب سے چلائی جاتی ہیں۔

سٹیٹمنٹ ٹرمینیٹر: ایک شناخت کنندہ جو کمپائلر کو بتاتا ہے کہ سٹیٹمنٹ ختم ہو گئی ہے۔C- لینگوئج میں سیمی کولن (;) بطور سٹیٹمنٹ ٹرمینیٹر استعمال ہوتا ہے۔

سٹرنگ: کریکٹر کا مجموعہ۔

سنٹیکس: ہر پروگرامنگ لینگوئج کے کچھ ابتدائی تعمیراتی عناصر اور پروگرام لکھنے کے کچھ اصول ہوتے ہیں۔ اصولوں کے اس سیٹ کو لینگوئج سنٹیکس (Language Syntax) کہتے ہیں۔

شارٹ سرکٹنگ: پورے ایکسپریشن پر کام کیے بغیر اوپریشن کا جواب نکالنا۔

شناخت کنندہ: ایک متغیر کا حوالہ دینے کے لیے استعمال کیا جانے والا نام۔

فنکشن کی باڈی: یہ ان سٹیٹمنٹس کا سیٹ ہوتا ہے۔ جنھیں چلا کر فنکشن مخصوص کام سرانجام دیتا ہے۔

فنکشن کو کال کرنا: اس فنکشن کو کنٹرول ٹرانسفر کرنا۔

فلوٹنگ پوائنٹ: ایک ڈیٹا ٹائپ جو چھ ہندسوں تک پر سائز رئیل کانسٹنٹ (Precise Real Constant) محفوظ کرنے کے لیے استعمال ہوتی ہے۔ اس کا سائز 4 بائٹس ہے۔

فارمیٹ سپیسفائرز: یہ ان پٹ آؤٹ پٹ اوپریشنز کے دوران ڈیٹا ٹائپ کا فارمیٹ بنانے کے لیے استعمال ہوتے ہیں۔

فنکشن سگنیچر: یہ فنکشن کی ان پٹ اور آؤٹ پٹ کی وضاحت کرتا ہے۔

فنکشن: سٹیٹمینٹس کا ایک بلاک جو ایک خاص کام سرانجام دیتا ہے۔

Getch() فنکشن: یہ صارف سے ایک کریکٹر لینے کے لیے استعمال ہوتا ہے اسے صرف کریکٹر ان پٹ دی جا سکتی ہے درج کیا گیا کریکٹر سکرین پر ظاہر نہیں ہوتا۔

کریکٹر: ایک ڈیٹا ٹائپ جس میں صرف کریکٹر محفوظ کیے جا سکتے ہیں۔ اس کا سائز بائٹ ہوتا ہے۔

کمنٹس: وہ سٹیٹمنٹس جو چلتی نہیں ہیں۔

کمپائلر: ایک ایسا سافٹ ویئر جو کسی ہائی لیول کی پروگرامنگ لینگوئج میں لکھے گئے کوڈ کو ایسے کوڈ میں تبدیل کر دیتا ہے جسے مشین سمجھ سکے۔

کمپیوٹر پروگرام: ایک خاص کام کو کرنے کے لیے انسان کی لکھی گئی ہدایات کی فہرست۔

اصطلاحات

کمپیوٹر پروگرامنگ: کمپیوٹر پروگرام کی ہدایات کو کمپیوٹر میں محفوظ کرنے کا عمل ۔

کنڈیشن: اس میں کوئی بھی درست ایکسپریشن آ سکتا ہے جیسے ارتھمیٹک ایکسپریشنز، ریلیشنل ایکسپریشنز، منطقی ایکسپریشنز یا ان سب کا مجموعہ ۔

کانسٹینٹس: وہ ڈیٹا جسے مزید پروسیسنگ کے لئے کمپیوٹر کی میموری میں محفوظ کیا جاتا ہے ۔

کنٹرول سٹیٹمنٹس: وہ پروگرام جو تسلسل کو کنٹرول کرتی ہے ۔

کی ۔ ورڈز: پروگرامنگ لینگوئج میں پہلے سے ڈیفائن کیے ہوئے الفاظ کی فہرست ۔

ہیڈر فائلز: وہ فائلز جن میں ڈیزائنرز نے موجودہ فنکشنز ڈیفائن کیے ہوں ۔

ہیڈر سیکشن: وہ سیکشن جس میں ہیڈر فائلز شامل کی جاتی ہیں ۔

لوپ سٹرکچر: یہ سٹیٹمنٹس کے ایک سیٹ کو بار بار دہرانے کے لیے استعمال ہوتا ہے ۔

مشروط سٹیٹمنٹس: وہ سٹیٹمینٹس جو شرائط کی بنا پر یہ فیصلہ کرنے میں مدد دیتی ہیں کے آگے کونسی سٹیٹمینٹس چلانی ہیں ۔

منطقی AND اوپریٹر: یہ true جواب دیتا ہے اگر دونوں طرف کے ایکسپریشنز true ہوں ۔

منطقی NOT اوپریٹر: یہ true جواب دیتا ہے اگر ایکسپریشن false ہو اور false جواب دیتا ہے اگر ایکسپریشن true ہو ۔

منطقی اوپریٹر: یہ بولین ایکسپریشنز پر اوپریشن انجام دیتے ہیں اور جواب میں بولین قیمت واپس کرتے ہیں ۔

مین فنکشن: یہاں سے پروگرام چلنا شروع ہوتا ہے ۔

ماڈولس اوپریٹر: ایک بائنری اوپریٹرز جو بائیں اوپرینڈ کو دائیں اوپرینڈ پر تقسیم کرتا ہے اور تقسیم کے بعد بچنے والی رقم واپس کرتا ہے ۔

متغیر کی ڈکلیریشن: متغیر کا نام اور ڈیٹا ٹائپ متعین کرنا ۔

متغیر کی انیشلائزیشن: پہلی مرتبہ ایک متغیر سے قیمت منسوب کرنا ۔

متغیرات: وہ کنٹینرز جن میں کانسٹینٹس یا قیمتیں محفوظ کی جاتی ہیں ۔

نیسٹڈ سلیکشن سٹرکچرز: مشروط سٹیٹمنٹس میں مشروط سٹیٹمنٹس

()printf فارمیٹڈ آؤٹ پٹ سکرین پر دکھانے کا بلٹ ان فنکشن ۔

()scanf: صارف سے فارمیٹڈ ان پٹ ریڈ کرنے والا C لینگوئج کا بلٹ ان فنکشن ۔

یونری اوپریٹرز: انھیں صرف ایک اوپرینڈ درکار ہوتا ہے ۔

یوزر ڈیفائنڈ فنکشنز: وہ فنکشنز جنھیں پروگرامر خود ڈیفائن کرے ۔

انڈیکس

(الف)
ANDاوپریٹر،39
آرگومینٹس،108
ارتھمیٹک اوپریٹرز،32
ارے کی ڈکلریشن، 79
ارے کی انیشلائزیشن 79،
انڈیکس،80
انٹیجر کانسٹینٹ،10
انٹیجر، 12
انٹی گریٹڈ ڈیویلپمنٹ انوائرنمنٹ،3
اوپریٹر،31
ان سائنڈ int،12
OR اوپریٹر،40
ارے کا سائز،79
ارے،78
اسائنمنٹ اوپریٹر،316
ایڈیٹر، 4
اسکیپ کریکٹر،29
ان پٹ، آؤٹ پٹ اوپریشنز،23
if سٹیٹمنٹس، 53
if-else سٹیٹمنٹس، 59
(ب)
بائنری اوپریٹر،41
بولین،12
بلٹ ان فنکشن،103
بار بار استعمال کی صلاحیت،104
(پ)
پیرامیٹرز،104
Printf' فنکشن،23
پروگرامر،2
پروگرامنگ انوائرنمنٹ،2
پروگرامنگ لینگوئجز،2

(ت)
ترجیح،41
(ج)
جمع کا اوپریٹر،35
(ڈ)
ڈیٹا سٹرکچر،78
ڈیٹا ٹائپ،116
(ذ)
ذخیرہ الفاظ،6
(ر)
رئیل کانسٹنٹ،10
ریشنل کریکٹرز،37
ریٹرن ویلیو،104
(س)
Scanf،26
سائنڈ int،12
سنگل لائن کمنٹ،8
سٹیٹمنٹ ٹرمینیٹر،29
سٹرنگ،12
سنٹیکس ایرر،5
سنٹیکس،5
(ش)
شناخت کنندہ،11
شرط53
(ض)
ضرب کا آپریٹرز،34
(ف)
فلوٹنگ پوائنٹ،12
فارمیٹ سپیسفائر،24
فنکشن کال،107
فنکشن ڈیفینیشن،105
فنکشن سگنیچر،105
فنکشن،103

(ک)
کریکٹر کانسٹینٹ،34
کنٹرول سٹیٹمنٹس،52
کمپیوٹر پروگرام،2
کمپیوٹر پروگرام،2
conio.h،28
کنسول،5
کانسٹینٹس،10
(گ)
getch()،28
(ل)
لنک سیکشن،6
لوپ سٹرکچر،83
(م)
منسوب کردہ کوڈ،53
مین فنکشن کی باڈی،7
مطلوبہ الفاظ،6
منطقی اوپریٹرز،39
مین فنکشن،7
مین فکشن 7
ماڈولس آپریٹر،36

جوابات

باب1:

سوال نمبر1: کثیرالانتخابی سوالات۔

1۔ ج	6۔ ج
2۔ الف	7۔ ب
3۔ ب	8۔ ب
4۔ ب	9۔ ب
5۔ الف	10۔ ج

سوال نمبر2: درست/غلط۔

1۔ درست

2۔ درست

3۔ غلط

4۔ غلط

5۔ درست

سوال نمبر3: کالم ملائیں۔

1۔	d
2۔	f
3۔	a
4۔	e
5۔	g
6۔	b
7۔	h
8۔	e

جوابات

باب2:

سوال نمبر 1: کثیر الانتخابی سوالات۔

1۔ د
2۔ ج
3۔ ج
4۔ ب
5۔ ب
6۔ د
7۔ ب
8۔ ب
9۔ ج
10۔ د

سوال نمبر 2: درست/غلط۔

1۔ غلط
2۔ درست
3۔ غلط
4۔ غلط
5۔ غلط

سوال نمبر 3: آؤٹ پٹ۔

1۔ 0 7 9
2۔ nn

nnn
n
nn /n/n nn/n t

3۔ 9
4۔ 5
5۔ 1

باب3:

سوال نمبر1: کثیر الانتخابی سوالات۔

1۔ الف 2۔ د 3۔ ج 4۔ الف 5۔ ب 6۔ د 7۔ الف 8۔ ج

سوال نمبر5: آؤٹ پٹ۔

1. a = 17, b = 10
2. Hope for the Best
3. 6 < 9 and N = N
4. a=50176, b=224, c=22, d=484
5. x = 16
   x = 16, y = 8, z = 9

جوابات

باب4:

سوال نمبر1: کثیرالانتخابی سوالات۔

1۔ ج
2۔ الف
3۔ الف
4۔ الف
5۔ ب
6۔ د
7۔ د
8۔ ج
9۔ الف
10۔ الف

سوال نمبر5: آؤٹ پٹ۔

1. Sum is 25

2. *

3. j = 50
j = 49
j = 48
i = 50

4. 4
16
36
64

5. 0.000000
0.000000
0.000000
2.640000
5.520000
11.959999

باب5:

سوال نمبر1: کثیرالانتخابی سوالات۔

1۔ ج
2۔ ب
3۔ د
4۔ ب
5۔ د
6۔ الف
7۔ د
8۔ الف

سوال نمبر5: آؤٹ پٹ۔

1۔ 20
2۔ 4 7 23
3۔ We got 260
4۔ 0.000000
0.000000