Programming Projects

1. Ans:

```
#include <stdio.h>

void dispense(int, int*, int*, int*);

void main()
{
	int dollars, tens = 0, fifs = 0, twens = 0;
	printf("\nEnter the amount desired in dollars (multiple of 10) - ");
	scanf("%d", &dollars);
	if((dollars % 10) == 0)
	{
		dispense(dollars, &tens, &twens, &fifs);
		printf("\nDollars - %d", dollars);
		if(tens != 0)
			printf("\n10s - %d", tens);
		if(twens != 0)
			printf("\n20s - %d", twens);
		if(fifs != 0)
			printf("\n50s - %d", fifs);
		printf("\n");
	}
	else
		printf("\nEntered amount is not a multiple of 10\n");
}

void dispense(int dollars, int *tens, int *twens, int *fifs)
{
	*fifs = dollars / 50;
	dollars = dollars % 50;
	*twens = dollars / 20;
	dollars = dollars % 20;
	*tens = dollars / 10;
	dollars = dollars % 10;
}
```

O/P

Enter the amount desired in dollars (multiple of 10) - 30
Dollars - 30
10s - 1
20s – 1

Enter the amount desired in dollars (multiple of 10) - 50
Dollars - 50
50s – 1

Enter the amount desired in dollars (multiple of 10) - 80
Dollars - 80
10s - 1
20s - 1
50s – 1

Enter the amount desired in dollars (multiple of 10) - 75
Entered amount is not a multiple of 10

2. Ans:

```
#include<stdio.h>
#include<math.h>

void cal_change(double ,int*, int*, int*, int*);

void main()
{
   int dollars, quaters = 0, dimes = 0, nickels = 0, pennies = 0;
   double amount_paid , amount_due, amount_change, coin_change;

   printf("Enter amount due:\n");
   scanf("%lf", &amount_due);

   printf("Enter amount paid:\n");
   scanf("%lf", &amount_paid);

   amount_change =  amount_paid - amount_due;
   dollars = floor(amount_change);
   coin_change = (amount_change - dollars) * 100;

   cal_change(coin_change, &quaters, &dimes, &nickels, &pennies);

   printf("\nChange in dollars - %d $", dollars);
   printf("\nQuarters: - %d", quaters);
   printf("\nDimes - %d", dimes);
   printf("\nNickels - %d", nickels);
   printf("\nPennies - %d", pennies);
}

void cal_change(double coin_change, int *quaters, int *dimes, int *nickels, int *pennies)
{
   int q = 1,d = 1,n = 1,p = 1;
   do
   {
                        if(coin_change>=25)
                        {
                        *quaters += q;
                        coin_change -= 25;
                        }
                        else if(coin_change>=10)
                        {
                                *dimes += d;
                                coin_change -= 10;
                        }
                        else if(coin_change>=5)
                        {
                                *nickels += n;
                                coin_change -= 5;
                        }
                        else
                        {
                                *pennies += p;
                                coin_change -= 1;
                        }
    }while(coin_change >= 0.9);
}
```

O/P

Enter amount due:
128.34
Enter amount paid:
200

Change in dollars: - 71 $
Quarters: - 2
Dimes: - 1
Nickels: - 1
Pennies: - 1

3. Ans:

```
#include <stdio.h>
#include <math.h>

void check(int, int*, int*, int*);
int digits_odd(int);
int isprime(int);
void main()
{
	int num, q1=0, q2=0, q3=0;

	printf("Enter a number-> ");
	scanf("%d", &num);

	check(num, &q1, &q2, &q3);

	if(q1 == 1)
		printf("\nThe number is a multiple of 7, 11, or 13.");
	else
		printf("\nThe number is not a multiple of 7, 11, or 13.");

	if(q2 == 1)
		printf("\nThe sum of the digits is odd.");
	else
		printf("\nThe sum of the digits is even.");

	if(q3 == 1)
		printf("\nThe number is a prime number.\n");
	else
		printf("\nThe number is not a prime number.\n");
}

void check(int num, int *q1, int *q2, int *q3)
{
	if(((num % 7) == 0) || ((num % 11) == 0) || ((num % 13) == 0))
		*q1 = 1;

	if(digits_odd(num))
		*q2 = 1;

	if(isprime(num))
		*q3 = 1;
}
```

```
int digits_odd(int num)
{
	int temp;
	int total = 0;
	if(num == 0)
		total = 0;
	else
	{
		while (num > 0)
		{
			temp = num % 10;
			total += temp;
		num = num / 10;
		}
	}

	if(total % 2 == 0)
		return 0;
	return 1;
}

int isprime(int n)
{
	int i, flag = 1;
	for(i = 2; i <= sqrt(n); i++)
	{
		if(n % i == 0)
		{
			flag = 0;
			break;
		}
	}
	return flag;
}
```

O/P

Enter a number-> 123

The number is not a multiple of 7, 11, or 13.
The sum of the digits is even.
The number is not a prime number.

Enter a number-> 7

The number is a multiple of 7, 11, or 13.
The sum of the digits is odd.
The number is a prime number.

4. Ans:

```
#include <stdio.h>
#include <math.h>
#define DIFFERENCE 0.005

void approximate_square_root(double, double, double*);

void main()
{
```

```
	double n, LG = 1.0, NG;

	n = 4;
	printf("\nNumber - %f", n);
	approximate_square_root(n, LG, &NG);
	printf("\nSquare root - %f", NG);

	n = 120.5;
	printf("\n\nNumber - %f", n);
	approximate_square_root(n, LG, &NG);
	printf("\nSquare root - %f", NG);

	n = 88;
	printf("\n\nNumber - %f", n);
	approximate_square_root(n, LG, &NG);
	printf("\nSquare root - %f", NG);

	n = 36.01;
	printf("\n\nNumber - %f", n);
	approximate_square_root(n, LG, &NG);
	printf("\nSquare root - %f", NG);

	n = 10000;
	printf("\n\nNumber - %f", n);
	approximate_square_root(n, LG, &NG);
	printf("\nSquare root - %f", NG);

	n = 0.25;
	printf("\n\nNumber - %f", n);
	approximate_square_root(n, LG, &NG);
	printf("\nSquare root - %f", NG);

	printf("\n");
}

void approximate_square_root(double N, double LG, double *NG)
{
	*NG = 0.5 * (LG + (N / LG));
	while(fabs(*NG - LG) > DIFFERENCE)
	{
		LG = *NG;
		*NG = 0.5 * (LG + (N / LG));
	}
}
```

O/P

Number - 4.000000
Square root - 2.000000

Number - 120.500000
Square root - 10.977249

Number - 88.000000
Square root - 9.380832

Number - 36.010000
Square root - 6.000833

Number - 10000.000000
Square root - 100.000000

Number - 0.250000
Square root - 0.500000

5. Ans:

```
#include <stdio.h>
#include <math.h>
#define RHO 1.23

void calc_drag_force(double*, double, double, double);

void main()
{
   int V;
   double CD, A, F;

   printf("\nEnter the area: ");
   scanf("%lf", &A);
   printf("\nEnter the drag co-efficient: ");
   scanf("%lf", &CD);

   printf("\n\nVelocity %9c Force", ' ');

   for(V = 0; V <= 40; V += 5)
   {
      calc_drag_force(&F, CD, A, V);
      printf("\n%d m/s %8c %.2f N", V, ' ', F);
   }
   printf("\n");
}

void calc_drag_force(double *F, double CD, double A, double V)
{
   *F = 0.5 * CD * A * RHO * pow(V, 2);
}
```

O/P

Enter the area: 100
Enter the drag co-efficient: 0.3

Velocity	**Force**
0 m/s	**0.00 N**
5 m/s	**461.25 N**
10 m/s	**1845.00 N**
15 m/s	**4151.25 N**
20 m/s	**7380.00 N**
25 m/s	**11531.25 N**
30 m/s	**16605.00 N**
35 m/s	**22601.25 N**
40 m/s	**29520.00 N**

6. Ans:

```
#include <stdio.h>
#include <math.h>
```

```
void do_next_op(char, double, double*);
void scan_data(char*, double*);

void main()
{
   double accumulator = 0, operand;
   char operator_symbol;

   printf("\n+ add");
   printf("\n- subtract");
   printf("\n* multiply");
   printf("\n/ divide");
   printf("\n^ power (raise left operand to power of right operand)");
   printf("\nq quit\n");

   scan_data(&operator_symbol, &operand);

   while(operator_symbol != 'q')
   {
      do_next_op(operator_symbol, operand, &accumulator);

      printf("result so far is %.1f", accumulator);
      scan_data(&operator_symbol, &operand);
   }
   printf("\nFinal result is %.1f\n", accumulator);
}

void do_next_op(char operator_symbol, double operand, double *accumulator)
{
   if(operator_symbol == '+')
      *accumulator += operand;
   else if(operator_symbol == '-')
      *accumulator -= operand;
   else if(operator_symbol == '*')
      *accumulator *= operand;
   else if(operator_symbol == '/')
   {
      if(operand != 0)
         *accumulator /= operand;
      else
         printf("divide by zero error\n");
   }
   else if(operator_symbol == '^')
      *accumulator = pow(*accumulator, operand);
   else
      printf("invalid operator\n");
}

void scan_data(char *operator_symbol, double *operand)
{
   printf("\n");
   scanf(" %c", operator_symbol);
   scanf(" %lf", operand);
}
```

O/P

+ add
- subtract
*** multiply**
/ divide
^ power (raise left operand to power of right operand)
q quit

+ 5.0
result so far is 5.0
^ 2
result so far is 25.0
/ 2.0
result so far is 12.5
q 0

Final result is 12.5

7. Ans:

```
#include <stdio.h>
#include <math.h>

double revenue(int);
void predict(double);

void main()
{
	double trillion = pow(10, 12);
	predict(trillion);
}

double revenue(int t)
{
	t -= 1984;
	double R = 203.265 * pow(1.071, t);
	return R;
}

void predict(double R)
{
	int t = 1984;
	FILE *output = fopen("out_5.7.txt", "w");

	fprintf(output, "Year %5c Revenue", ' ');

	while(revenue(t) <= R)
	{
		fprintf(output, "\n%d %5c %.3f", t, ' ', revenue(t));
		t++;
	}
	fclose(output);
}
```

O/P

out_5.7.txt

8. Ans:

```
#include <stdio.h>

void get_checksum(int, int*);
void main()
{
		char ch,ec;
		int sum, checksum;
		while(1){
		printf("Enter a single-line message ending with a period (Or only period to stop)>");
		scanf("%c", &ec);
		if(ec == '.')
				break;
		else{
		sum = (int)ec;
		   do{
		      scanf("%c", &ch);
		      sum += ((int)ch);
		   }while(ch != '.');
		get_checksum(sum, &checksum);
		printf("checksum :- %c (%d)\n", (char)checksum, checksum);
		}
		scanf("%c", &ec);
		}
}

void get_checksum(int sum, int *checksum)
{
		sum %= 64;
		*checksum = sum + (int)' ';
}
```

O/P

Enter a single-line message ending with a period (Or only period to stop)>PSPD.
checksum :- E (69)
Enter a single-line message ending with a period (Or only period to stop)>C.
checksum :- Q (81)
Enter a single-line message ending with a period (Or only period to stop)>.

9. Ans:

```
#include <stdio.h>
#include <math.h>

void approximate(int, double*);

void main()
{
   int x = 1;
   double approx, difference = 1;

   while(difference >= 0.000001)
   {
		approximate(x, &approx);
		difference = fabs(exp(1) - approx);
		x++;
   }
```

```
   printf("\nFinal approximation of e using expression:- %.7f at x = %d\n", approx,x-1);
   printf("\nThe value of e calculated by the exp function:- %.7f", exp(1));
}

void approximate(int x, double *approx)
{
		double temp = ((2.0 * x) + 1.0) / ((2.0 * x) - 1.0);
		*approx = pow(temp, x);
}
```

O/P

Final approximation of e using expression:- 2.7182828 at x = 476

The value of e calculated by the exp function:- 2.7182818