Tutorial & Laboratory

Programming & Data Structure: CS11001/19001

Section - 4/D

DO NOT POWER ON THE MACHINE

Department of Computer Science and Engineering I.I.T. Kharagpur

Spring Semester: 2013 - 2014 (**03.04.2014**)

Download

Download the file date030414.pdf from Programming & Data Structures ... of

http://cse.iitkgp.ac.in/~goutam

View the file using the command acroread & or xpdf &

Laboratory Test II: 10^{th} April, 2013

- Lab Test I + 1-D array, recursive function, string, 2-D array, structures, data types, lists, sorting, searching.
- Do not carry any book or note book.
- Time: Two hours.
- Marks: 35

The test will start after the tutorial.

A Polynomial Over $\mathbb{R}$

A single-variable polynomial over the real numbers is written as

$$p(x) = a_0 + a_1 x + a_2 x^2 + \cdots + a_{n-1} x^{n-1},$$

where all a_i's are real numbers and x is the formal variable.

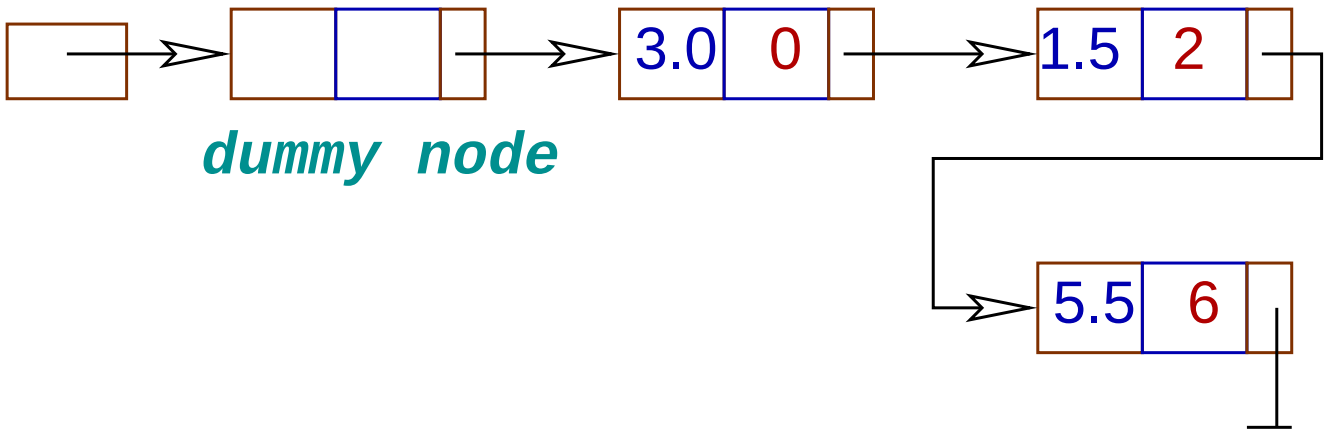
Data in a Polynomial

Data in a polynomial is a finite sequence of (a_i, n_i) pairs, where each pair corresponds to the term $a_i x^{n_i}$.

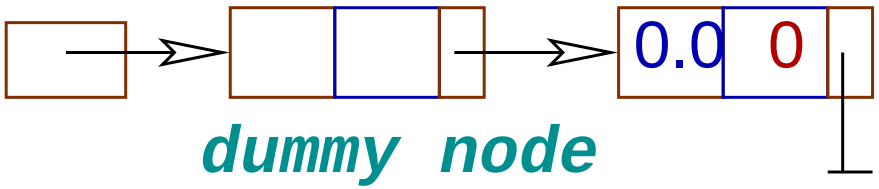
Representation of Polynomial

The polynomial may be stored as an ordered linked-list of terms.

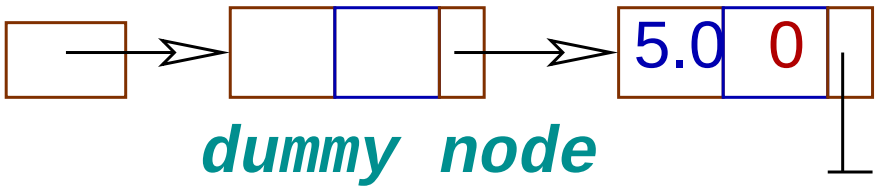
$$3.0 + 1.5x^2 + 5.5x^6$$



Zero Polynomial



Constant Polynomial



Type Definition

```
typedef struct term {
        float coeff;
        int exp;
        struct term *next;
} term, *poly;
```

Operations on a Polynomial

Following are a few operations on a polynomial: `initPoly()`, `readPoly()`, `printPoly()`, `addPoly()`, `subPoly()`, `multPoly()`, `diffPoly()`, `evalPoly()` etc.

Assignment XVI

Write a C program to implement the following functions. Down load the `main16.c` file which contains `int main()`, `poly initPoly()` which returns a zero polynomial with a dummy node, and `int isZero(poly p)` which tests whether p is a zero polynomial. [Marks: $5 + 5 + 5 + 5 + 5 + 5$]

Task I

Write a C function `term *makeMono(float a, int n)` that returns a monomial, a pointer to a single `term` containing ax^n i.e. (a, n). There is no dummy node. [Marks: 5]

Task II

Write a C function `void insertTerm(poly p, term *tP)` that takes a polynomial `p` and a `term tP`. It inserts the term in proper place in the ordered-list if there is no term with the exponent of `tP`. If it is a zero polynomial, the zero term present in `p` is removed. [Marks: 5]

Task III

Write a C function `void readPoly(poly p)` that reads a sequence of exponent, coefficient pairs, constructs monomials corresponding to each pair using `makeMono()`, and inserts it in `p` using `insertTerm()`.
The input stream is terminated by a negative exponent. [Marks: 5]

Note

Input corresponding to $3.0 + 1.5x^2 + 5.5x^6$ may be 2 1.5 6 5.5 0 3.0 -1 or 0 3.0 2 1.5 6 5.5 -1 etc.

Task IV

Write a C function `void printPoly(poly p)` that prints the polynomial of `p` in understandable form. [Marks: 5]

Sample IO

Parameter: `(5.5, 6) (-1.5, 2) (3.0,0)`
Output : `5.50x^6- 1.50x^2+ 3.00`

Task V

Write a C function `poly addPoly(poly p, poly q)` adds two polynomials p and q and returns the polynomial sum. It should not modify p, q. [Marks: 5]

Sample IO

```
Input: 7 5.0 4 -3.0 2 4.0 0 -1.0 -1
Output :
p: 5.00x^7- 3.00x^4+ 4.00x^2- 1.00
Input: -2.0 3 6.0 7 -3.0 8 8.0 -1
Output :
q: 8.00x^8- 3.00x^7+ 6.00x^3- 2.00x^2
Output :
sum: 8.00x^8+ 2.00x^7- 3.00x^4+ 6.00x^3
           + 2.00x^2- 1.00
```

Task VI

Write a C function `poly multPoly(poly p, poly q)` multiplies two polynomials p and q and returns the product polynomial. It should not modify p, q. [Marks: 5]

Sample IO

```
Input: 7 5.0 4 -3.0 2 4.0 0 -1.0 -1
Output :
p: 5.00x^7- 3.00x^4+ 4.00x^2- 1.00
Input: 0 -2.0 3 6.0 7 -3.0 8 8.0 -1
Output :
q: 8.00x^8- 3.00x^7+ 6.00x^3- 2.00x^2
Output : 40.00x^15-15.00x^14-24.00x^12+
      9.00x^11+62.00x^10-22.00x^9- 8.00x^8
      -15.00x^7+ 6.00x^6+24.00x^5- 8.00x^4
      - 6.00x^3+ 2.00x^2
```

Sample IO - 1

Input: 0 3 -1
Output: 3.00
Input: 0 4 -1
Output: 4.00
Output: 7.00
Output: 12.00

Sample IO - 2

Input: 0 3 -1
Output: 3.00
Input: 2 2.5 0 -1.0 -1
Output: 2.50x^2- 1.00
Output: 2.50x^2+ 2.00
Output: 7.50x^2- 3.00

Sample IO - 3

Input: 0 0.0 -1
Output: 0.00
Input: 2 2.5 0 -1.0 -1
Output: 2.50x^2- 1.00
Output: 2.50x^2- 1.00
Output: 0.00

Sample IO - 4

Input: 2 2.5 0 -1.0 -1
Output: `2.50x^2- 1.00`
Input: 0 0.0 -1
Output: 0.0@
Output: `2.50x^2- 1.00`
Output: `0.00`

Sample IO - 5

```
Input: 7 5.0 4 -3.0 2 4.0 0 -1.0 -1
Output : 5.00x^7- 3.00x^4+ 4.00x^2- 1.00
Input: 0 -2.0 3 6.0 7 -3.0 8 8.0 -1
Output : 8.00x^8- 3.00x^7+ 6.00x^3- 2.00
Output: 8.00x^8+ 2.00x^7- 3.00x^4+
          6.00x^3+ 4.00x^2- 3.00
Output: 40.00x^15-15.00x^14-24.00x^12+
          9.00x^11+62.00x^10-12.00x^9-
          8.00x^8-25.00x^7+24.00x^5+
          6.00x^4- 6.00x^3- 8.00x^2+ 2.00
```

Submission by ftp

```
$ ftp 10.5.17.186
Connected to 10.5.17.186.
220---------- Welcome to Pure-FTPd ----
220-You are user number 1 of 50 allowed.
220-Local time is now 07:54. ....... 21.
220-IPv6 connections ........
220 ... disconnected .. inactivity.
Name (10.5.17.186:...): pds
```

Submission by ftp

```
331 User pds OK. Password required
Password: pds04
230-User pds has group access to:  pds
230 OK. Current restricted directory is /
Remote system type is UNIX.
Using binary mode to transfer files.
ftp> cd assignment16
250 OK. Current directory is /assignment16
```

Submission by ftp

```
ftp> put D0616.c
local: D0616.c remote: D0616.c
200 PORT command successful
150 Connecting to port 47093
226-File successfully transferred
226 0.001 seconds .. 39.00 Kbytes ..
27 bytes sent in 0.00 secs (1098.6 kB/s)
ftp> bye
21-Goodbye. .....
221 Logout.
$
```