

<https://www.facebook.com/hasan.cse91><https://github.com/hasancse91><https://bd.linkedin.com/in/abdullah-al-hasan-376030b1><https://plus.google.com/u/0/+AbdullahAlHasanCSE91>

হাসানের রফখাতা

<https://hellohasan.com/>

পোস্টটি পড়া হয়েছে 3,221 বার



লিংকড লিস্ট – ১ [Singly Linked List Create & Print in C]

January 4, 2017

<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/>

/ Hasan Abdullah

<https://hellohasan.com/author/hasan-cse91/> / Singly Linked List

<https://hellohasan.com/category/data-structure/linked-list/singly-linked-list/>

আমার এই ব্লগ বা অন্য যে কোন ব্লগের পোস্টের শেষে সাধারণত পরের পোস্টের লিংক দেয়া থাকে। ধরো এই ব্লগের ডেটা স্ট্রাকচার সিরিজের অ্যারের উপর লেখা প্রথম পোস্টটি তুমি পড়ে শেষ করলো। পোস্টের শেষে অ্যারের দ্বিতীয় পোস্টের লিংক দেয়া আছে। দ্বিতীয় পোস্টের শেষে আবার তৃতীয় পোস্টের লিংক দেয়া আছে। তোমার কী মনে হয়, অ্যারের উপর লেখা সবগুলো পোস্টই কি একটার পর একটা লেখা হয়েছে? না...!

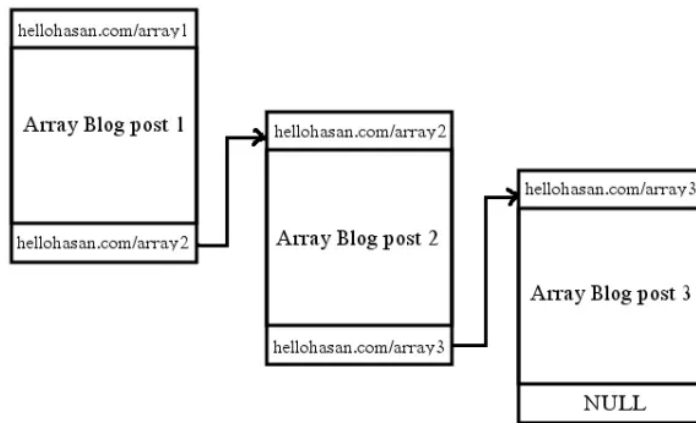
ধরো অ্যারের প্রথম পোস্ট লিখার পরে ইচ্ছা করল অ্যালগরিদম সিরিজের কোন একটা লেখা পোস্ট করতে। তো সেই পোস্ট করলাম। এরপর আবার ইচ্ছা করল অ্যাক্সেসড ডেভেলপমেন্টের উপর একটা লেখা দেই। এরপর ইচ্ছা করলো অ্যারের দ্বিতীয় লেখাটা লিখার। দ্বিতীয় লেখাটা পোস্ট করার সময় প্রথম পোস্টের শেষে দ্বিতীয় পোস্টের লিংক যোগ করে দেই। দ্বিতীয় পোস্টের শেষে কিন্তু কোন লিংক যোগ করি নাই। কারণ অ্যারের তৃতীয় পোস্ট এখনো লেখা হয় নাই। এর মানে দাঁড়াচ্ছে, আমার কোনো একটা সিরিজের লিস্টে নতুন কোনো পোস্ট লেখা হলে সর্বশেষ পোস্টের শেষে নতুন পোস্টের লিংক জুড়ে দিচ্ছি।

আপনার জেলার ৫ ওয়াক্ত নামাজের শুরু ও শেষের সময় এবং সেহরি ও ইফতারের

সময়সূচী জানতে ইন্সটল করুন আমাদের অ্যাক্সেসড অ্যাপ

(<https://play.google.com/store/apps/details?id=theoaktroop.appoframadan>)

আবার একই সিরিজের ৫ টা পোস্ট লেখা হলে যদি মনে হয় ৩ নাম্বার পোস্টের পরে নতুন একটা পোস্ট হলে ভাল হয়। তাহলে কী করবো? ৩ নাম্বার পোস্টের শেষে নতুন লেখা পোস্টের লিংক জুড়ে দিব। নতুন পোস্টের শেষে যোগ করব ৪ নাম্বার পোস্টের লিংক। তাহলে কিন্তু খুব সহজেই আমার পোস্টগুলোর একটার পর একটার ক্রমটা ঠিক মত হ্যান্ডেল করা গেল। আবার কোনো কারণে মনে হলো ২ নাম্বার পোস্টটা ভুল হয়েছে। এটা রাখা ঠিক হবে না। তাহলে ১ নাম্বার পোস্টের শেষে লিংক করব ৩ নাম্বার পোস্ট। তাহলে কিন্তু ২ নাম্বার পোস্টটা আমার লিংক করা লিস্ট থেকে ছিটকে পড়ে গেল।



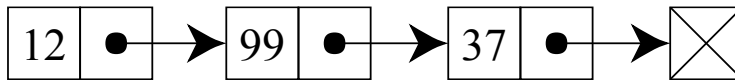
লিংকড লিস্টের উদাহরণ

উপরের এই মহা পেচাপেচি (!) বুঝতে কোনো সমস্যা আছে? যদি বুঝে গিয়ে থাকো তাহলে তোমার লিংকড লিস্ট নামক মহান ডেটা স্ট্রাকচার বুঝা হয়ে গেছে! অভিনন্দন তোমাকে! 😊

ডেটা স্ট্রাকচার বা অ্যালগরিদম যা-ই বলো না কেন প্রথম কঠিন কাজটা হচ্ছে আইডিয়াটা বুঝা, অনুধাবন করা। তুমি যদি concept-টা বুঝে ফেলো তাহলে আজ না হোক কাল ঠিকই কোড করতে পারবে। এ জন্যেই বললাম উপরের উদাহরণের আইডিয়া বুঝে গিয়ে থাকলে তোমার লিংকড লিস্ট বুঝার কঠিন পাটটা শেষ। বাকিটুকু হচ্ছে সহজ কাজ, কোড লিখা। তার আগে চলো কেতাবি ঢঙ্গে লিংকড লিস্ট নিয়ে আরো কিছু আলোচনা করা যাক।

বিসিএস, GRE, ব্যাংক জব, শিক্ষক নিবন্ধন সহ যে কোন চাকুরির পরীক্ষার প্রস্তুতির জন্য
ডাউনলোড করুন Editorial Word অ্যাপ্লিকেড অ্যাপ
(<https://play.google.com/store/apps/details?id=megaminds.dailyeditorialword>)

জ্ঞানগুরু উইকিপিডিয়ার ভাষ্য মতে “**A linked list is a linear collection of data elements, called nodes, each pointing to the next node by means of a pointer. It is a data structure consisting of a group of nodes which together represent a sequence.**” অর্থাৎ লিংকড লিস্ট হচ্ছে কিছু নোডের লিনিয়ার কালেকশন, যেই নোডগুলো একেকটা তার পরেরটাকে পয়েন্টারের মাধ্যমে পয়েন্ট করে। এটা একটা ডেটা স্ট্রাকচার, যেখানে নোডগুলো একত্রে একটা সিকোয়েন্স তৈরি করে থাকে।



লিংকড লিস্টের উদাহরণ. Wikipedia থেকে নেয়া

মাথার উপর দিয়ে গেল তাই না? গেলে যাক! মূল ব্যাপারটা যেহেতু বুঝেই গেল এত গুরুগম্ভীর আলোচনাকে পাতা না দিলেও চলবে।

কোডিং পার্টে ঢুকান আগে তোমার কিছু বিষয়ে নলেজ থাকতে হবে। যেমনঃ অ্যারে, স্ট্রাকচার, পয়েন্টার ও হালকা পাতলা রিকার্সন। অ্যারের উপর বিস্তারিত লেখাগুলো পাওয়া যাবে এখানে (<https://hellohasan.com/category/data-structure/array/>)। রিকার্সনের প্রাথমিক ধারণা পেতে পারো আমার ব্লগের রিকার্সন সিরিজ থেকে (<https://hellohasan.com/tag/recursion/>)। বাকি টপিকগুলো ব্লগে এখনো লিখি নাই। গুগল করে শিখে ফেলতে পারো।

ফিরে যাই পোস্টের শুরুতে উল্লেখ করা উদাহরণে। সেখানে আমরা একটা পোস্টের সাথে আরেকটা পোস্টকে লিংক করেছিলাম। পরের পোস্টের সাথে লিংক করেছিলাম সেই পোস্টের অ্যাড্রেসের মাধ্যমে। পোস্টের শেষে উল্লেখ করা অ্যাড্রেসে ক্লিক করলে সেই পোস্টটি পড়া যায়। অর্থাৎ কিছু ডেটা পাওয়া যায়। অ্যাড্রেসের কাজ কিন্তু সেরেফ লিংক করা। মূল জিনিস বা মূল উদ্দেশ্য কিন্তু ব্লগের লেখাটা পড়া তাই না? তাহলে উদাহরণের এলিমেন্টগুলোকে দুই ভাগ করি। প্রথম ভাগ বা মূল জিনিস হচ্ছে ব্লগ পোস্টের কন্টেন্ট বা লেখাগুলো। আর দ্বিতীয় অংশ হচ্ছে পরের পোস্টের ওয়েব অ্যাড্রেস যা শুধু লিংক করার জন্য ব্যবহৃত হবে। মনে করো প্রতিটা ব্লগ পোস্টে ঢুকলে তুমি বড় করে একটা নাম্বার দেখতে পাবে। এটাই ধরে নাও মূল ডেটা। আর এই নাম্বারের পরে ছোট করে পরের পোস্টের অ্যাড্রেস দেখতে পাবে।

তুমি যেহেতু স্ট্রাকচার জানো তাই বলছি। এই নাম্বার আর অ্যাড্রেসকে একটা ডেটা হিসেবে কিভাবে উল্লেখ করা যায়? খুব সহজেই একটা স্ট্রাকচার বানিয়ে ফেলতে পারো এভাবেঃ

```
1 struct blog_post
2 {
3     int number;
4     string address;
5 };
```

তাহলে blog_post হচ্ছে একটা স্ট্রাকচার। এটাতে রয়েছে ব্লগের মূল ডেটা (number) এবং পরের পোস্টের অ্যাড্রেস (address). এবার এই blog_post টাইপের ডেটার কয়েকটা ভেরিয়েবল বানিয়ে ফেলি।

```

1 .
2 blog_post blog_post1, blog_post2, blog_post3;
3 .

```

তিনটা ব্লগ পোস্ট তৈরি করা হয়েছে। আমরা যদি প্রথমটার address ভেরিয়েবলে দ্বিতীয় পোস্টের লিংকটা রাখতে পারি, দ্বিতীয় পোস্টের address ভেরিয়েবলে তৃতীয় পোস্টের লিংক রাখতে পারি তাহলে কিন্তু প্রথমটার অ্যাড্রেসে ক্লিক করলে দ্বিতীয় পোস্ট, দ্বিতীয় পোস্টের নিচে থাকা অ্যাড্রেসে ক্লিক করলে তৃতীয় পোস্টটি পড়তে পারব। যেহেতু ৩ টাই মাত্র পোস্ট। তাই তৃতীয় পোস্টের অ্যাড্রেসে আপাতত NULL রেখে দিতে পারি। কারণ পরে আর কোন পোস্ট নাই। নতুন কোনো পোস্ট যোগ হলে তার লিংকটা রেখে দিব blog_post3 এর address এ। আর নতুনটার address এ রেখে দিব NULL. এই আইডিয়াটাই লিংকড লিস্ট। এই আইডিয়া কাজে লাগিয়ে আমরা সত্যিকারের লিংকড লিস্ট ইমপ্লিমেন্ট করবো।

এক কথায় লিংকড লিস্টের সংজ্ঞা বলতে চাইলে বলা যায় লিংকড লিস্ট হচ্ছে কতগুলো স্ট্রাকচারের একটা লিস্ট। যেই স্ট্রাকচারগুলোর মধ্যে এক বা একাধিক ডেটা থাকতে পারে। এবং পরবর্তী স্ট্রাকচারের মেমরি অ্যাড্রেস থাকে। অন্যান্য ডেটা স্ট্রাকচারের মত লিংকড লিস্ট ডেটা স্ট্রাকচারেরও কিছু কমন অপারেশন রয়েছে। সেগুলো আমরা আস্তে আস্তে কভার করবো।

Operations of Linked List

- Create linked list
- Traverse
- Counting the list item
- Print the full list
- Search an item on list
- Insert a new item on list
- Delete an item from list
- Concatenate two linked list

Types of Linked List:

1. Linear Singly Linked List
2. Circular Linked List
3. Doubly Linked List
4. Circular Doubly Linked List

এই পোস্টে প্রথম টাইপের লিংকড লিস্ট নিয়েই আলোচনা করা হবে। পরবর্তী পোস্টে বাকিগুলো নিয়ে আলোকপাত করার ইচ্ছা আছে।

Problem Definition

তোমাকে একটা প্রোগ্রাম লিখতে হবে যেটা ডায়নামিক্যালি একটা int টাইপের লিস্ট তৈরি করতে পারে। অর্থাৎ ইউজার আগে থেকে ইনপুট দিবে না যে সে কয়টা এলিমেন্টের লিস্ট তৈরি করতে চায়। ইউজার হয়ত কখনো ৫ টা সংখ্যার লিস্ট তৈরি করবে, আবার কখনো ৫০০০ সংখ্যার লিস্ট তৈরি করবে। শর্ত হচ্ছে সে যতটা সংখ্যার লিস্ট তৈরি করবে ঠিক ততটুকু মেমরিই দখল করা যাবে। শুরুতেই তুমি অনেক বড় একটা অ্যারে ডিক্লেয়ার করে রাখলে হবে না। এক্ষেত্রে মেমরি খুব সীমিত। তাই প্রয়োজনের অতিরিক্ত ১ বাইটও খরচ করা যাবে না। Problem টি সলভ করতে হবে Linked List এর মাধ্যমে।

Solution

লিংকড লিস্ট যেহেতু একটা স্ট্রাকচারের লিস্ট। তাই শুরুতেই একটা স্ট্রাকচার বানিয়ে ফেলি:

Create node by structure for Linked List

```
1 struct linked_list
2 {
3     int number;
4     struct linked_list *next;
5 };
```

ডেটা হিসেবে এখানে আছে number. তোমাদের প্রয়োজন অনুসারে এখানে যতগুলো দরকার ডেটা নিতে পারো। next হচ্ছে এই linked_list টাইপের স্ট্রাকচারের একটা পয়েন্টার ভেরিয়েবল। যে কিনা এই টাইপের একটা স্ট্রাকচারের মেমরি অ্যাড্রেস সংরক্ষণ করতে পারে।

main function এর উপরে, এই linked_list স্ট্রাকচারের একটা global variable declare করি node নাম দিয়ে এভাবে:

```
1 .
2 typedef struct linked_list node;
3 .
```

typedef কী?

typedef এমন একটা keyword যার মাধ্যমে তুমি যে কোন টাইপের নতুন নামকরণ করতে পারবে। উদাহরণ দিলে ব্যাপারটা পরিষ্কার হবে।

Typedef Example in C

```
1 {
2 .
3     typedef char Book[100];
4     Book book1;
5     scanf("%s", book1);
6     printf("%s",book1);
7 .
8 }
```

char টাইপের একটা অ্যারে ডিক্লেয়ার করা হয়েছে Book[100] লিখে। এর শুরুতে typedef কীওয়ার্ডটা বসানো হয়েছে। এর পরের লাইনে দেখা, Book টাইপের একটা ভেরিয়েবল ডিক্লেয়ার করা হয়েছে। অর্থাৎ নতুন কোন ডেটাটাইপ না, কিন্তু আমাদের বুঝার সুবিধার্থে কোন একটা ভেরিয়েবলকেই আমরা ডেটাটাইপের মত করে ব্যবহার করতে পারি। বা Type define করে দিতে পারি।

ফিরে আসি লিংকড লিস্টে। প্রবলেমটা সলভ করার জন্য আমাদের procedure হচ্ছে, main function এ node এর একটা পয়েন্টার ভেরিয়েবল (head) তৈরি করা। যে কিনা লিস্টের প্রথম আইটেমের মেমরি অ্যাড্রেস সংরক্ষণ করবে। এরপর main function থেকে create ফাংশন কল করা হবে। প্যারামিটার হিসাবে পাঠানো হবে head-কে। এই head এর সাথে লিস্টের পরের আইটেমগুলো একটার পর একটা যুক্ত হতে থাকবে। head তৈরির কাজটা করা যায় এভাবে:

Create head node of a Linked List

```
1 .
2 node *head; //node টাইপের ভেরিয়েবলের মেমরি অ্যাড্রেস সংরক্ষণ করবে head
3 head = (node *) malloc(sizeof(node)); //node টাইপের ভেরিয়েবলের মেমরি অ্যাড্রেস assign করা হয়েছে
4 .
```

malloc কী?

Dynamic memory allocation এর জন্য এই ফাংশনটি ব্যবহৃত হয়। আমরা একটা int type এর ভেরিয়েবল ডিক্লেয়ার করতে পারি int a; লিখে। এতে মেমরির যে কোন একটা অ্যাড্রেসে a এর জন্য মেমরি অ্যালোকেট করা হয়। কিন্তু কখনো যদি সরাসরি ভেরিয়েবল ডিক্লেয়ার না করে ভেরিয়েবলের মেমরি ডিক্লেয়ার করার দরকার হয় তখন আমরা malloc ব্যবহার করতে পারি।

malloc example in C

```

1 {
2     int *a;
3     a = (int *) malloc (sizeof(int));
4     printf("Memory address is %d\n",a);
5
6     scanf("%d", a); //input to address "a". "a" is the memory address. So no need to us
7     printf("%d", *a); //"a" is memory address. but "*a" is the value of address "a"
8 }

```

আমাদের লিংকড লিস্টের ক্ষেত্রে প্রতিটা নতুন নতুন node লিস্টের সাথে জুড়ে দেয়ার সময় malloc ব্যবহার করে নতুন নোডের জন্য memory allocate করে হবে। আর মেমরি অ্যাড্রেসটা আগের নোডের next (মেমরি অ্যাড্রেস) variable এ অ্যাসাইন করে দিলেই তৈরি হয়ে যাবে লিংকড লিস্ট।

আগে বলে দেয়া procedure অনুযায়ী আমাদের head নোড তৈরি করা হয়ে গেছে। এখন create(head) ফাংশন কল করে এই head এর সাথে লেজ জুড়ে দেয়ার কাজ করতে হবে।

Create a Linked List

Create function of a Linked List

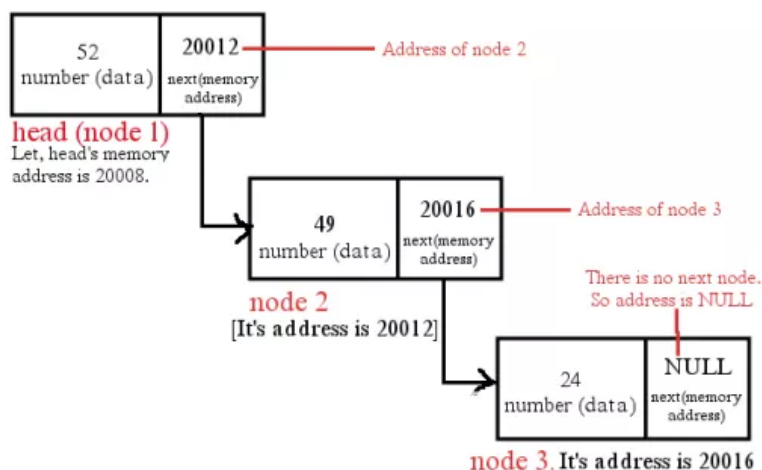
```

1 void create(node *myList)
2 {
3     printf("Input a number. (Enter -99999 at end)\n");
4
5     scanf("%d", &myList->number);
6
7     if(myList->number== -99999)
8         myList->next = NULL;
9     else
10    {
11        myList->next = (node *) malloc(sizeof(node));
12        create(myList->next);
13    }
14 }

```

এই ফাংশনের প্যারামিটার হিসেবে পাঠানো হয়েছে একটা memory address. আর অ্যাড্রেসটা হচ্ছে node টাইপের একটা স্ট্রাকচারের। লিংকড লিস্টের ক্ষেত্রে প্রায় সব কাজই কিন্তু মেমরি অ্যাড্রেস নিয়ে করতে হবে। কোনো আইটেম add করা, delete করা ইত্যাদি অপারেশনগুলো করার উপায় এই মেমরি অ্যাড্রেস ধরে ধরে।

create() একটি রিকার্সিভ ফাংশন। এর base case হচ্ছে -99999 ইনপুট হওয়া। অর্থাৎ কখনো যদি -99999 ইনপুট করা হয় তাহলে ধরে নেয়া হবে লিস্টটা আর বড় হবে না। -99999 এর আগের সংখ্যাটিই লিস্টের সর্বশেষ আইটেম। base case সত্যি হলে current node এর pointer ভেরিয়েবল next = NULL করে দেয়া হবে। এতে বুঝা যাবে এর পরে আর কোন আইটেম নাই, এটিই সর্বশেষ আইটেম।



লিংকড লিস্টের উদাহরণ

যদি base case সত্য না হয়, তাহলে **myList->next = (node *)**

malloc(sizeof(node)); এর মাধ্যমে নতুন একটা node এর জন্য মেমরি দখল করা হলো।

এরপর সেই মেমরি অ্যাড্রেসটা দিয়ে আবার create(myList->next); কল করা হলো। যতক্ষণ -99999 ইনপুট না হবে ততক্ষণ এই রিকার্সিভ কল চলতেই থাকবে। এরই মাধ্যমে আমাদের লিস্ট তৈরির কাজ শেষ হলো।

Print the linked list

পুরো লিস্টটা প্রিন্ট করতে চাইলে main function থেকে print function কল করতে হবে।

প্যারামিটার হিসাবে থাকবে লিস্টের শুরুর node বা head.

Print the whole Linked List

```
1 void print(node *myList)
2 {
3     printf("%d ", myList->number);
4
5     if(myList->next == NULL)
6         return;
7
8     print(myList->next);
9 }
```

এটাও একটা recursive function. ফাংশন বডিতে ঢুকেই current node এর number-টা print করে দিবে। পরের নোডের অ্যাড্রেস রাখা আছে next নামক pointer variable এ। যদি এতে NULL পাওয়া যায় তার মানে হচ্ছে print করার মত এর পরে আর কোনো নোড নাই। তাই return করার মাধ্যমে ফাংশনের কাজ শেষ করা হচ্ছে। অন্যথায় পরের নোডের অ্যাড্রেস দিয়ে আবারো print() কল হচ্ছে। এভাবে পুরো লিস্টটি প্রিন্ট করা হচ্ছে।

Size of linked list

লিস্টে কতগুলো আইটেম আছে সেটা জানার জন্য এই ফাংশনটা কল করা যায়:

Size of a Linked List

```
1 int countListItem(node *myList)
2 {
3     if(myList->next == NULL)
4         return 0;
5
6     return (1 + countListItem(myList->next));
7 }
```

প্রিন্ট করার মতই। তাই আর ব্যাখ্যায় গেলাম না।

পুরো কোডটি পাওয়া যাবে আমার **গিটহাব রিপোজিটরিতে**

(<https://github.com/hasancse91/data-structures/blob/master/Source%20Code/Linked%20List.c>)।


লিংকড লিস্ট – ২ [Create, insert, delete, search]


(<https://hellohasan.com/2017/01/08/%E0%A6%B2%E0%A6%BF%E0%A6%82%E0%A6%95%E0%A6%A1-%E0%A6%B2%E0%A6%BF%E0%A6%B8%E0%A7%8D%E0%A6%9F-linked-list-create-insert-delete-search/>)


পর্বে আলোচনা করা হয়েছে আরো কিছু ব্যাসিক অপারেশন।


লিংকড লিস্টের আইডিয়াটা ক্লিয়ার হবে নিজ হাতে কোড করলে। এখানকার কোড কপি পেস্ট করো না। প্রয়োজনে দেখে দেখে টাইপ করে কোড করো। পুরো ব্যাপারটা ক্লিয়ার হয়ে যাবে। আজ এ পর্যন্তই। কোথাও কোন ভুল-ত্রুটি চোখে পড়লে জানাও। এছাড়া পোস্ট সম্পর্কে যে কোন মতামতও জানাতে পারো। ধন্যবাদ।


Share this:


 Facebook 72
(https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=facebook&nb=1)


 WhatsApp (whatsapp://send?text=%E0%A6%B2%E0%A6%BF%E0%A6%82%E0%A6%95%E0%A6%A1%20%E0%A6%B2%E0%A6%BF%E0%A6%B8%E0%A7%8D%E0%A6%9F%20-%20%E0%A7%A7%20%5BSingly%20Linked%20List%20Create%20%26%20Print%20in%20C%5Dhttps%3A%2F%2Fhellohasan.com%2F2017%2F01%2F04%2F%25e0%25a6%25b2%25e0%25a6%25bf%25e0%25a6%2582%25e0%25a6%2595%25e0%25a6%25a1-%25e0%25a6%25b2%25e0%25a6%25bf%25e0%25a6%25b8%25e0%25a7%258d%25e0%25a6%259f-linked-list-create-print-size%2F)


 Skype (https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=skype&nb=1)


 Google (https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=google-plus-1&nb=1)


 LinkedIn 2
(https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=linkedin&nb=1)


 Pocket (https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=pocket&nb=1)


 Reddit (https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=reddit&nb=1)

 Twitter (https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=twitter&nb=1)

 Telegram (https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=telegram&nb=1)

 Pinterest (https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=pinterest&nb=1)

 Print (https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/#print)

 Email (https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?share=email&nb=1)

Related

- স্ট্যাক: বহুল ব্যবহৃত ডেটা স্ট্রাকচার (https://hellohasan.com/2016/10/31/%e0%a6%b8%e0%a7%8d%e0%a6%9f%e0%a7%8d%e0%a6%af%e0%a6%be%e0%a6%95-stack-data-structure/) October 31, 2016 In "স্ট্যাক - Stack"

মার্জ সর্ট অ্যালগরিদম - Merge Sort Algorithm (https://hellohasan.com/2016/10/22/%e0%a6%ae%e0%a6%be%e0%a6%b0%e0%a7%8d%e0%a6%9f-%e0%a6%85%e0%a7%8d%e

লিংকড লিস্ট - ২ [Singly Linked List Create, insert, delete, search in C] (https://hellohasan.com/2017/01/08/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e

0%a6%af%e0%a6%be%e0%
a6%b2%e0%a6%97%e0%a6%
%b0%e0%a6%bf%e0%a6%a
6%e0%a6%ae-merge-sort-
algorithm/)
October 22, 2016
In "সিটিং অ্যালগরিদম"

0%a6%b8%e0%a7%8d%e0%
a6%9f-linked-list-create-
insert-delete-search/)
January 8, 2017
In "Singly Linked List"

Tagged C - সি (<https://hellohasan.com/tag/c/>)

DEQUE বা DOUBLE-ENDED QUEUE
([HTTPS://HELLOHASAN.COM/2016/12/30/DEQUE-DOUBLE-ENDED-QUEUE-CPP-STL/](https://hellohasan.com/2016/12/30/deque-double-ended-queue-cpp-stl/))

লিংকড লিস্ট – ১ [SINGLY LINKED LIST
CREATE, INSERT, DELETE, SEARCH IN
C]
([HTTPS://HELLOHASAN.COM/2017/01/08/%E0%A6%B2%E0%A6%BF%E0%A6%82%E0%A6%95%E0%A6%A1-%E0%A6%B2%E0%A6%BF%E0%A6%B8%E0%A7%8D%E0%A6%9F-LINKED-LIST-CREATE-INSERT-DELETE-SEARCH/](https://hellohasan.com/2017/01/08/%E0%A6%B2%E0%A6%BF%E0%A6%82%E0%A6%95%E0%A6%A1-%E0%A6%B2%E0%A6%BF%E0%A6%B8%E0%A7%8D%E0%A6%9F-LINKED-LIST-CREATE-INSERT-DELETE-SEARCH/))

6 thoughts on “লিংকড লিস্ট – ১ [Singly Linked List Create & Print in C]”



জিকে ইমন says:

March 4, 2017 at 10:13 pm

(<https://hellohasan.com/2017/01/04/%E0%A6%B2%E0%A6%BF%E0%A6%82%E0%A6%95%E0%A6%A1-%E0%A6%B2%E0%A6%BF%E0%A6%B8%E0%A7%8D%E0%A6%9F-LINKED-LIST-CREATE-PRINT-SIZE/#comment-317>)

মেমোরি এসাইন মানে কি ভাই?

Reply (<https://hellohasan.com/2017/01/04/%E0%A6%B2%E0%A6%BF%E0%A6%82%E0%A6%95%E0%A6%A1-%E0%A6%B2%E0%A6%BF%E0%A6%B8%E0%A7%8D%E0%A6%9F-LINKED-LIST-CREATE-PRINT-SIZE/?replytocom=317#respond>)



Hasan Abdullah (<https://hellohasan.com>) says:

March 4, 2017 at 10:15 pm

(<https://hellohasan.com/2017/01/04/%E0%A6%B2%E0%A6%BF%E0%A6%82%E0%A6%95%E0%A6%A1-%E0%A6%B2%E0%A6%BF%E0%A6%B8%E0%A7%8D%E0%A6%9F-LINKED-LIST-CREATE-PRINT-SIZE/#comment-318>)

পুরো সেনটেন্সটা বলেন। নাহলে বুঝতে পারছি না

Reply (<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?replytocom=318#respond>)



A.K.M. Masud says:

May 31, 2017 at 2:56 am

(<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/#comment-343>)

please write the code in c++

Reply (<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?replytocom=343#respond>)



Hasan Abdullah (<https://hellohasan.com>) says:

May 31, 2017 at 8:48 pm

(<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/#comment-344>)

If you can understand the code, you can write it yourself. Data structure is a concept. It's not language specific domain. Learn the concept and implement it any language you want.

Good luck!

Reply (<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?replytocom=344#respond>)



musabbir ahmed khan says:

July 3, 2017 at 6:56 pm

(<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/#comment-384>)

Onek onek thnx

Reply (<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?replytocom=384#respond>)



Zahirul Islam says:



September 9, 2017 at 10:31 am

(<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/#comment-445>)

thanks sir

Reply (<https://hellohasan.com/2017/01/04/%e0%a6%b2%e0%a6%bf%e0%a6%82%e0%a6%95%e0%a6%a1-%e0%a6%b2%e0%a6%bf%e0%a6%b8%e0%a7%8d%e0%a6%9f-linked-list-create-print-size/?replytocom=445#respond>)

Leave a Reply

Your email address will not be published. Required fields are marked *

Comment

Name *

Email *

Website

Post Comment

Hit counter

- 251,449 hits

Post Categories

☐ অ্যান্ড্রয়েড অ্যাপ ডেভেলপমেন্ট
(<https://hellohasan.com/category/android-tutorial/>) (15)

☐ Android Library
(<https://hellohasan.com/category/android-tutorial/popular-android-library/>) (7)

☐ Development Tools
(<https://hellohasan.com/category/android-tutorial/android-development-tools/>) (2)

☐ Essential Topics
(<https://hellohasan.com/category/android-tutorial/essential-android-tutorial/>) (6)

☐ কম্পিউটার সায়েন্স ও আইটি
(<https://hellohasan.com/category/computer-science-it-industry/>) (11)

☐ প্রযুক্তি
(<https://hellohasan.com/category/technology/>) (3)

☐ প্রোগ্রামিং
(<https://hellohasan.com/category/programming/>) (6)

☐ অনলাইন জাজ সিরিজ
(<https://hellohasan.com/category/online-programming-judge-series/>) (13)

☐ ডেটা স্ট্রাকচার-Data Structure
(<https://hellohasan.com/category/data-structure/>) (21)

☐ সাধারণ আলোচনা
(<https://hellohasan.com/category/data-structure/data-structure-introduction/>) (1)

- অ্যারে – Array
(<https://hellohasan.com/category/data-structure/array/>) (2)
- স্ট্যাক – Stack
(<https://hellohasan.com/category/data-structure/stack/>) (3)
- কিউ – Queue
(<https://hellohasan.com/category/data-structure/queue/>) (2)
- লিংকড লিস্ট – Linked List
(<https://hellohasan.com/category/data-structure/linked-list/>) (6)
 - Singly Linked List
(<https://hellohasan.com/category/data-structure/linked-list/singly-linked-list/>) (2)
 - Doubly Linked List
(<https://hellohasan.com/category/data-structure/linked-list/doubly-linked-list/>) (2)
 - Circular Linked List
(<https://hellohasan.com/category/data-structure/linked-list/circular-linked-list/>) (2)
- ট্রি – Tree
(<https://hellohasan.com/category/data-structure/tree/>) (7)
 - ব্যাসিক কনসেপ্ট
(<https://hellohasan.com/category/data-structure/tree/tree-basic-concept/>) (2)
 - বাইনারি সার্চ ট্রি – BST
(<https://hellohasan.com/category/data-structure/tree/binary-search-tree-bst/>) (5)
- অ্যালগরিদম – Algorithm
(<https://hellohasan.com/category/algorithm/>) (9)

- সাধারণ আলোচনা
(<https://hellohasan.com/category/algorithm/algorithm-introduction/>) (2)
- সার্টিং অ্যালগরিদম
(<https://hellohasan.com/category/algorithm/searching-algorithm/>) (2)
- সার্টিং অ্যালগরিদম
(<https://hellohasan.com/category/algorithm/sorting-algorithm/>) (4)
- গ্রাফ অ্যালগরিদম
(<https://hellohasan.com/category/algorithm/graph-algorithm/>) (1)
- সংখ্যা পদ্ধতি সিরিজ
(<https://hellohasan.com/category/number-system-conversion-series/>) (11)
- অনুপ্রেরণা
(<https://hellohasan.com/category/inspiration/>) (3)
- জন সচেতনতা
(<https://hellohasan.com/category/public-awareness/>) (6)

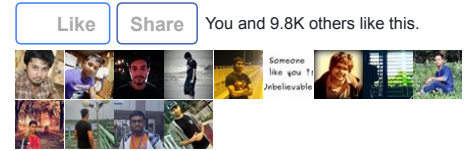
Facebook



Me on Facebook
(<https://www.facebook.com/hasan.cse91>)

Facebook Page of my Blog

(<https://www.facebook.com/HasanerRafkhata/>)



সাম্প্রতিক জনপ্রিয় পোস্টসমূহ

 (<https://hellohasan.com/2017/01/15/%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%bf-%e0%a6%a1%e0%a7%87%e0%a6%9f%e0%a6%be-%e0%a6%b8%e0%a7%8d%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%be%e0%a6%95%e0%a6%9a%e0%a6%be%e0%a6%b0-%e0%a7%a7-basic-concept/>)

ড্রি ডেটা স্ট্রাকচার - ১ [Basic Concept]


(<https://hellohasan.com/2017/01/15/%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%bf-%e0%a6%a1%e0%a7%87%e0%a6%9f%e0%a6%be-%e0%a6%b8%e0%a7%8d%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%be%e0%a6%95%e0%a6%9a%e0%a6%be%e0%a6%b0-%e0%a7%a7-basic-concept/>)



(<https://hellohasan.com/2016/09/05/computer-science-job-opportunities-in-bangladesh/>)

প্রোগ্রামিং ছাড়াও CSE
গ্র্যাজুয়েটদের আছে অনেক
চাকুরি
(<https://hellohasan.com/2016/09/05/computer-science-job-opportunities-in-bangladesh/>)



 (<https://hellohasan.com/2017/06/03/software-engineer-preparation-and-experience/>)

সফটওয়্যার ইঞ্জিনিয়ার হবার
জন্য আমার প্রস্তুতি ও গত দেড়
মাস চাকুরির অভিজ্ঞতা
(<https://hellohasan.com/2017/06/03/software-engineer-preparation-and-experience/>)

(<https://hellohasan.com/2017/01/15/%e0%a6%9f%e>



0%a7%8d%e0%a6%b0%e0%a6%
%bf-
%e0%a6%b8%e0%a7%8d%e0%
a6%9f%e0%a7%8d%e0%a6%b0
%e0%a6%be%e0%a6%95%e0%
a6%9a%e0%a6%be%e0%a6%b0
-applications-classification/)

ট্রি ডেটা স্ট্রাকচার - ২

[Applications and Classification]

(<https://hellohasan.com/2017/01/15/%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%bf-%e0%a6%b8%e0%a7%8d%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%be%e0%a6%95%e0%a6%9a%e0%a6%be%e0%a6%b0-applications-classification/>)



(<https://hellohasan.com/2017/10/01/android-retrofit-get-post-method-different-network-layer/>)

Android এ Retrofit

ব্যবহার করে GET ও POST

রিকোয়েস্ট [different

network layer] - ২

(<https://hellohasan.com/2017/10/01/android-retrofit-get-post-method-different-network-layer/>)



(<https://hellohasan.com/2017/01/15/%e0%a6%ac%e0%a6%be%e0%a6%87%e0%a6%a8%e0%a6%be%e0%a6%b0%e0%a6%bf-%e0%a6%b8%e0%a6%be%e0%a6%b0%e0%a7%8d%e0%a6%9a-%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%bf-binary-search-tree-bst/>)

(<https://hellohasan.com/2017/01/15/%e0%a6%ac%e0%a6%be%e0%a6%87%e0%a6%a8%e0%a6%be%e0%a6%b0%e0%a6%bf-%e0%a6%b8%e0%a6%be%e0%a6%b0%e0%a7%8d%e0%a6%9a-%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%bf-binary-search-tree-bst/>)

ট্রি ডেটা স্ট্রাকচার - ৩

[বাইনারি সার্চ ট্রি - BST]

(<https://hellohasan.com/2017/01/15/%e0%a6%ac%e0%a6%be%e0%a6%87%e0%a6%a8%e0%a6%be%e0%a6%b0%e0%a6%bf-%e0%a6%b8%e0%a6%be%e0%a6%b0%e0%a7%8d%e0%a6%9a-%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%bf-binary-search-tree-bst/>)



(<https://hellohasan.com/2017/07/16/android-app-development-guideline/>)

Android App ডেভেলপমেন্ট
গাইড লাইন

(<https://hellohasan.com/2017/07/16/android-app-development-guideline/>)



(<https://hellohasan.com/2016/10/07/%e0%a6%a1%e0%a7%87%e0%a6%9f%e0%a6%be-%e0%a6%b8%e0%a7%8d%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%be%e0%a6%95%e0%a6%9a%e0%a6%be%e0%a6%b0-%e0%a6%aa%e0%a6%b0%e0%a6%bf%e0%a6%9a%e0%a7%9f/>)
ডেটা স্ট্রাকচার কী ও কেন?

(<https://hellohasan.com/2016/10/07/%e0%a6%a1%e0%a7%87%e0%a6%9f%e0%a6%be-%e0%a6%b8%e0%a7%8d%e0%a6%9f%e0%a7%8d%e0%a6%b0%e0%a6%be%e0%a6%95%e0%a6%9a%e0%a6%be%e0%a6%b0-%e0%a6%aa%e0%a6%b0%e0%a6%bf%e0%a6%9a%e0%a7%9f/>)



(<https://hellohasan.com/2016/10/20/%e0%a6%ac%e0%a6%be%e0%a6%87%e0%a6%a8%e0%a6%be%e0%a6%b0%e0%a6%bf-%e0%a6%b8%e0%a6%be%e0%a6%b0%e0%a7%8d%e0%a6%9a-binary-search-algorithm/>)

বাইনারি সার্চ অ্যালগরিদম -

Binary Search

Algorithm

(<https://hellohasan.com/2016/10/20/%e0%a6%ac%e0%a6%be%e0%a6%87%e0%a6%a8%e0%a6%be%e0%a6%b0%e0%a6%bf-%e0%a6%b8%e0%a6%be%e0%a6%b0%e0%a7%8d%e0%a6%9a-binary-search-algorithm/>)

a6%be%e0%a6%b0%e0%
%a6%bf-
%e0%a6%b8%e0%a6%
be%e0%a6%b0%e0%a7
%8d%e0%a6%9a-
binary-search-
algorithm/)



(<https://hellohasan.com/2016/08/15/8-barriers-to-overcome-when-learning-to-code/>)

প্রোগ্রামিং শেখার সময় জয়
করতে হবে ৮ টি প্রতিবন্ধকতা
(<https://hellohasan.com/2016/08/15/8-barriers-to-overcome-when-learning-to-code/>)

Custom Ad: App of Ramadan 2017



(<https://goo.gl/mCYFRh>)

সাহরি-ইফতার ও সারা বছরের নামাজের
সময়সূচী জানার জন্য ব্যবহার করুন আমার
টিমের ডেভেলপ করা Android App!
(<https://goo.gl/mCYFRh>)

Ad – 1



(<https://goo.gl/hzOJkR>)

Ad – 2

ডোমেইন হোস্টিং সার্ভিসের সেরা প্যাকেজগুলো
দিচ্ছে Techno Haat!

(<https://goo.gl/mTGZLe>)



(<https://goo.gl/mTGZLe>)



Proudly powered by WordPress (<https://wordpress.org/>) | Theme: Amadeus (<http://themeisle.com/themes/amadeus/>) by Themeisle.