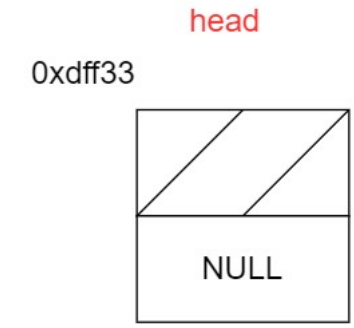
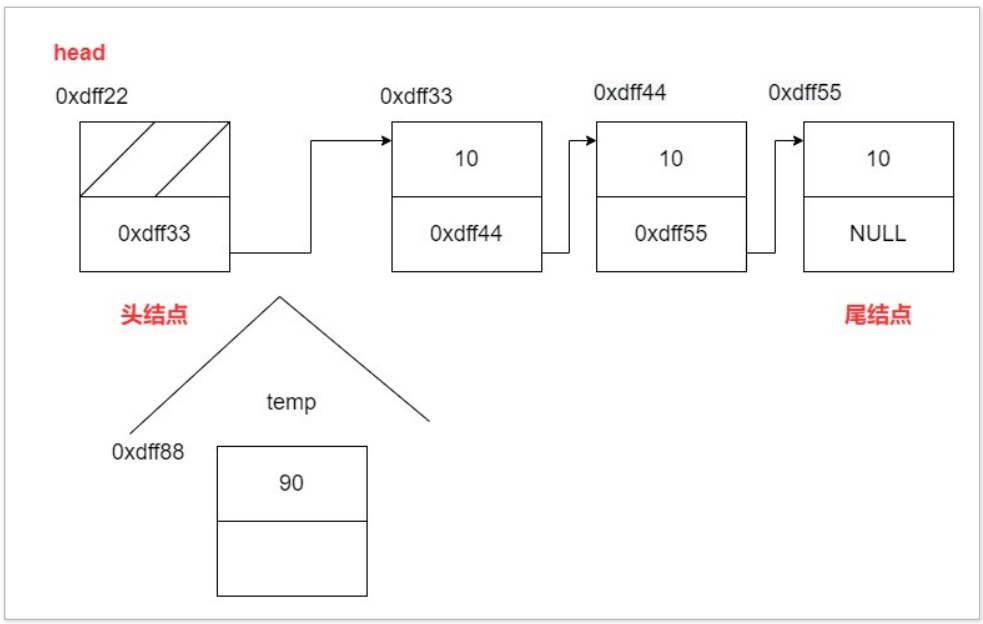
2.2 单向链表之创建, 插入, 输出_物联网 / 嵌入式工程师 - 慕课网

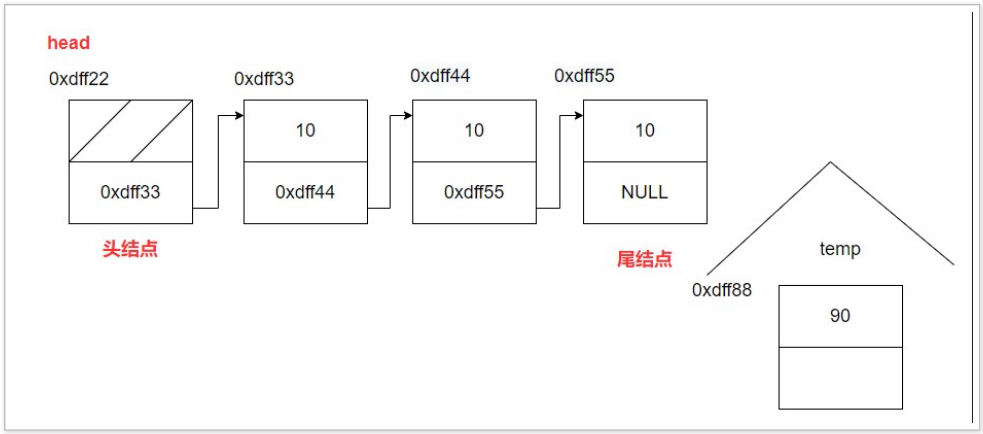
> 慕课网慕课教程 2.2 单向链表之创建, 插入, 输出涵盖海量编程基础技术教程，以图文图表的形式，把晦涩难懂的编程专业用语，以通俗易懂的方式呈现给用户。

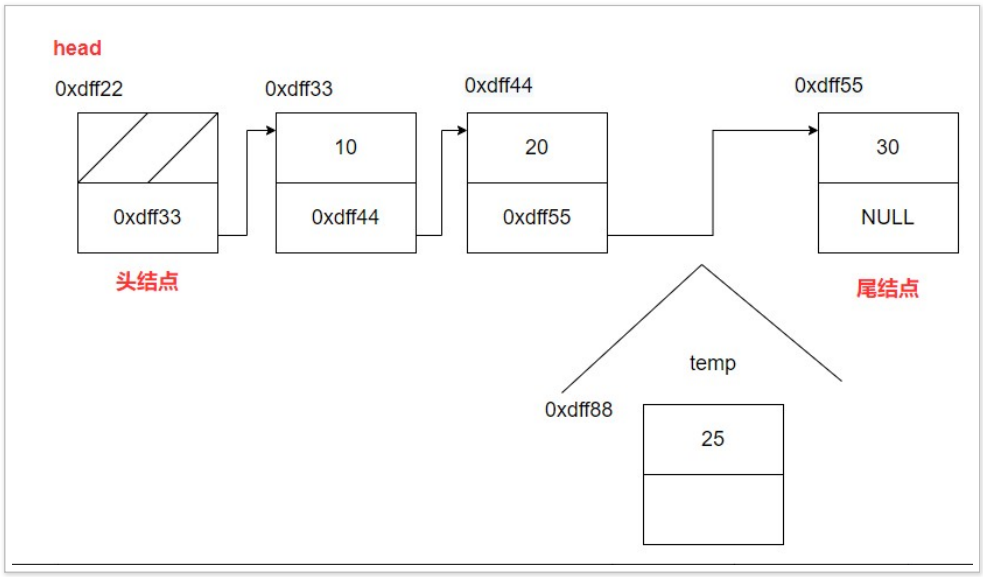
2. 单向链表之创建, 插入, 输出



- **插入**
  - 头插法：每次都在头结点 head 后插入 temp 结点。







head.h

```
#ifndef _HEAD_H_
#define _HEAD_H_

#include <stdio.h>
#include <string.h>
#include <stdlib.h>

typedef int datatype_t;

typedef struct node
{
    datatype_t data;
    struct node *next;
}linknode_t;

extern linknode_t *create_empty_linklist();
extern void insert_head_linklist(linknode_t *head,datatype_t data);
extern void printf_data_linklist(linknode_t *head);
extern void insert_tail_linklist(linknode_t *head,datatype_t data);
extern void insert_order_linklist(linknode_t *head,datatype_t data);
#endif
```

linklist.c

```
#include "head.h"


linknode_t *create_empty_linklist()
{
        linknode_t *head = NULL;



        head = (linknode_t *)malloc(sizeof(linknode_t));
```

```
        if(NULL == head)
        {
                printf("malloc is fail!\n");
                return NULL;
        }

        memset(head,0,sizeof(linknode_t));

        head->next = NULL;

        return head;
}



void insert_head_linklist(linknode_t *head,datatype_t data)
{

        linknode_t *temp = NULL;
        temp = (linknode_t *)malloc(sizeof(linknode_t));


        temp->data = data;

        temp->next = head->next;
        head->next = temp;
        return ;
}


void printf_data_linklist(linknode_t *head)
{
        linknode_t *p = head;

        while(p->next != NULL)
        {
                printf("%d ",p->next->data);
                p = p->next;
        }
        printf("\n");

        return ;
}

void insert_tail_linklist(linknode_t *head,datatype_t data)
{

        linknode_t *temp = NULL;
        temp = (linknode_t *)malloc(sizeof(linknode_t));

        temp->data = data;


        linknode_t *p = head;

        while(p->next != NULL)
        {
                p = p->next;
        }


        temp->next = p->next;
        p->next = temp;
        return ;
}

void insert_order_linklist(linknode_t *head,datatype_t data)
{

        linknode_t *temp = NULL;
        temp = (linknode_t *)malloc(sizeof(linknode_t));

        temp->data = data;

        linknode_t *p = head;

        while(p->next != NULL && data > p->next->data)
        {
                p = p->next;
        }

        temp->next = p->next;
        p->next = temp;

        return ;
}
```

main.c

```
#include "head.h"

int main()
{
        linknode_t *head = NULL;
        int n = 0,i = 0,ret = 0;
        datatype_t data;
        head = create_empty_linklist();

        printf("please input you want insert data number : ");
        scanf("%d",&n);

        printf("please input %d data : ",n);
        for(i = 0;i < n;i++)
        {
                scanf("%d",&data);


                insert_order_linklist(head,data);
        }

        printf_data_linklist(head);
        return 0;
}
```

```
please input you want insert data number : 10 20 30 40 50
10 20 30 40 50
```

创建一个单向链表, 把 1,5,3,7,9 无序数据要求按从大到小的方式利用 **** 有序插入的方式插入链表, 并输出。

全文完