Pertemuan 15

Manajemen Memori dan Linked List

Mata Kuliah : Algoritma & Pemrograman

Dosen : Tessy Badriyah, SKom., MT., PhD.

Pembahasan

- Konsep Manajemen Memori dalam program bahasa C
  - Penggunaan fungsi malloc(), calloc() dan free()
- Struktur Data Senarai Berantai (Linked List)
  - Single linked list
  - Double linked list

MEMORY MANAGEMENT

Mata Kuliah : Algoritma & Pemrograman

Dosen : Tessy Badriyah, SKom., MT., PhD.

Alokasi memori secara dinamis

- Dalam C, ukuran pasti dari sebuah array tidak diketahui sampai saat decompile.
- Sehingga seringkali ukuran dari array terlalu banyak (berlebihan) atau tidak cukup daripada yang diperlukan.
- Alokasi Memori secara dinamis (Dynamic memory allocation) memungkinkan program untuk mendapatkan space memori pada saat program dijalankan, atau merelease atau melepaskan alokasi dari memori jika tidak lagi diperlukan
- Untuk alokasi memori secara dinamis, digunakan standart library "stdlib.h" allocation.

4. Library Function under stdlib.h

- Function : malloc()
- Penggunaan fungsi : Mengalokasikan ukuran byte yang diperlukan dan mengembalikan byte pointer pertama dari ruang yang dialokasikan.
- Function : calloc()
- Penggunaan fungsi : Mengalokasikan ruang untuk elemen array, menginisialisasinya dengan nol dan mengembalikan pointer ke memori
- Function : free()
- Penggunaan fungsi : Mendealokasikan ruang yang dialokasikan sebelumnya (me-release memori)
- Function : realloc()
- Penggunaan fungsi : Merubah ukuran dari ruang yang dialokasikan sebelumnya

malloc()

- Nama dari malloc merupakan singkatan dari "memory allocation".
- Fungsi malloc() memesan blok memori dari ukuran yang sudah ditentukan dan mengembalikan pointer dari ruang yang dialokasikan.
- Syntax of malloc()

```
ptr = (cast-type*) malloc(byte-size)
```

- Contoh penggunaan :

```
ptr = (int*) malloc(100 * sizeof(int));
```

- Disini, ptr merupakan variable bertipe pointer. Fungsi malloc() mengambalikan pointer ke area memori dengan ukuran type. Jika ruang yang dipesan tidak mencukupi maka pointer akan dikembalikan dengan nilai NULL pointer.

calloc()

- Nama calloc merupakan singkatan dari "contiguous allocation".
- Perbedaan antara malloc() dan calloc() adalah bahwa, malloc() mengalokasikan blok memori tunggal sedangkan calloc() mengalokasikan beberapa blok memori dimana masing-masing memiliki ukuran yang sama dan semuanya diset awal sama dengan nol.
- Syntax of calloc()

```
ptr = (cast-type*)calloc(n, element-size);
```

- Perintah ini akan mengalokasikan contiguous space dalam memory untuk array yang terdiri dari 25 elemen.
- For example:

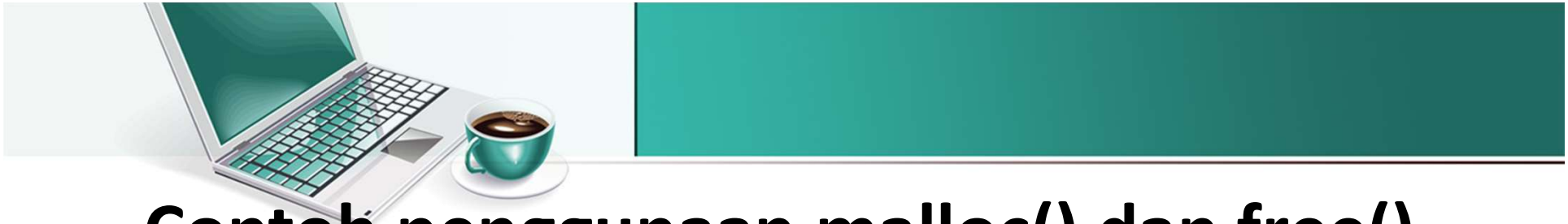
```
ptr = (float*) calloc(25, sizeof(float));
```

free()

- Alokasi memori yang dibuat secara dinamis dengan calloc() atau malloc() tidak dapat dibebaskan secara otomatis.
- Untuk membebaskan alokasi memori, kita gunakan free()
- Sintak dari free() :

```
free(ptr);
```

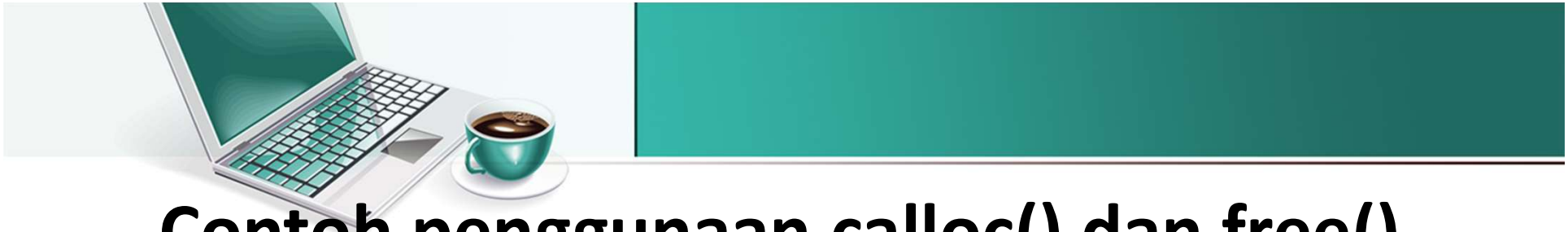
- Perintah free diatas akan membebaskan alokasi memori yang ditunjuk oleh variable pointer ptr.



Contoh penggunaan malloc() dan free()

- Buat program untuk menghitung total jumlah n buah elemen yang dimasukkan oleh user.
- Dengan menggunakan alokasi memori secara dinamis menggunakan fungsi malloc()

```
#include <stdio.h>
#include <stdlib.h>
int main()
{ int num, i, *ptr, sum = 0;
  printf("Enter number of elements: ");
  scanf("%d", &num);
  ptr = (int*) malloc(num * sizeof(int));  //memory allocated using malloc
  if(ptr == NULL)
  { printf("Error! memory not allocated.");
    exit(0);
  }
  printf("Enter elements of array: ");
  for(i = 0; i < num; ++i)
  { scanf("%d", ptr + i);
    sum += *(ptr + i);
  }
  printf("Sum = %d", sum);
  free(ptr);
  return 0;
}
```



Contoh penggunaan calloc() dan free()

- Buat program untuk menghitung total jumlah n buah elemen yang dimasukkan oleh user.
- Dengan menggunakan alokasi memori secara dinamis menggunakan fungsi calloc()

```
#include <stdio.h>
#include <stdlib.h>
int main()
{ int num, i, *ptr, sum = 0;
  printf("Enter number of elements: ");
  scanf("%d", &num);
  ptr = (int*) calloc(num, sizeof(int));
  if(ptr == NULL)
  { printf("Error! memory not allocated.");
    exit(0);
  }
  printf("Enter elements of array: ");
  for(i = 0; i < num; ++i)
  { scanf("%d", ptr + i);
    sum += *(ptr + i);
  }
  printf("Sum = %d", sum);
  free(ptr);
  return 0;
}
```

realloc()

- Jika alokasi memori sebelumnya tidak cukup atau lebih daripada yang diperlukan, maka kita bisa merubah ukuran dari alokasi memori menggunakan fungsi realloc().
- Sintak dari realloc()

```
ptr = realloc(ptr, newsize);
```

- Disini, variable pointer ptr direalokasikan dengan ukuran newsize.

Contoh penggunaan realloc()

```
#include <stdio.h>
#include <stdlib.h>
int main()
{ int *ptr, i , n1, n2;
  printf("Enter size of array: ");
  scanf("%d", &n1);
  ptr = (int*) malloc(n1 * sizeof(int));
  printf("Address of previously allocated memory: ");
  for(i = 0; i < n1; ++i)
     printf("%u\t",ptr + i);
  printf("\nEnter new size of array: ");
  scanf("%d", &n2);
  ptr = realloc(ptr, n2);
  for(i = 0; i < n2; ++i)
     printf("%u\t", ptr + i);
  return 0;
}
```

LINKED LIST

Mata Kuliah : Algoritma & Pemrograman

Dosen : Tessy Badriyah, SKom., MT., PhD.

Tujuan Pembelajaran

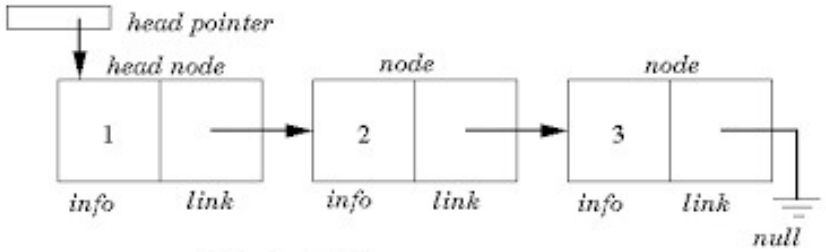
- Memahami konsep linked list
- Dapat menyelesaikan persoalan dengan menggunakan struktur data linked list

Konsep Linked List

- Linked list adalah suatu bentuk struktur yang memiliki elemen (item) yang menunjuk ke suatu alamat tertent memori
- Memiliki 2 tipe utama :
  - Single linked list
  - Double linked list
-

Single Linked List

- Ilustrasi single linked list



A Linked List

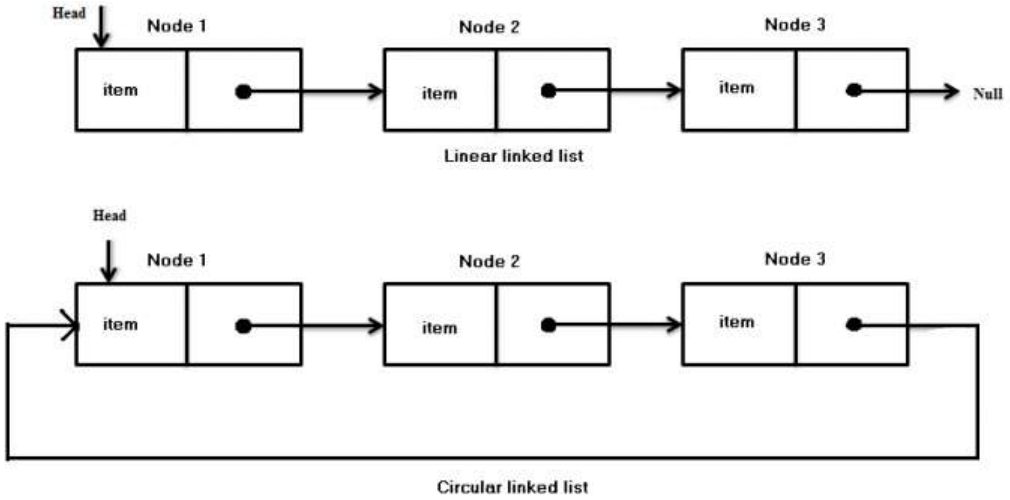
```
#include<stdlib.h>
#include<stdio.h>
#include<conio.h>
struct list_el {
   int val;
   struct list_el * next;
};
typedef struct list_el item;
```

Contoh program Single linked list

```
main() {
  item * curr, * head;
  int i;
  head = NULL;
  for(i=1;i<=10;i++) {
    curr = (item *)malloc(sizeof(item));
    curr->val = i;
    curr->next  = head;
    head = curr;
  }
  curr = head;
  while(curr) {
    printf("%d\n", curr->val);
    curr = curr->next ;
  }
  getch();    }
```

Circular single linked list

- Circular single linked list sama dengan single linked list, hanya pointer pada tail akan menunjuk ke head.



Soal 1)

Implementasikan bentuk circular single linked list ke dalam program.

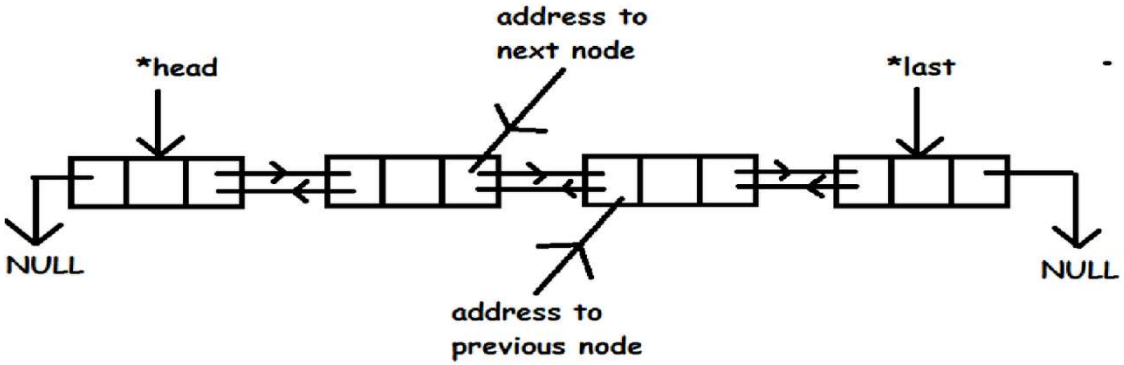
Soal 2

- Modifikasi program dengan linked list berikut sehingga bisa menampilkan bilangan dengan urutan kebalikannya
- Menggunakan single linked list

```
main() {
  item * curr, * head;
  int i;
  head = NULL;
  for(i=1;i<=10;i++) {
    curr = (item *)malloc(sizeof(item));
    curr->val = i;
    curr->next  = head;
    head = curr;
  }
  curr = head;
  while(curr) {
    printf("%d\n", curr->val);
    curr = curr->next ;
  }
  getch();    }
```

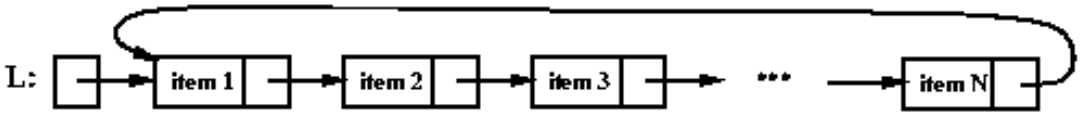
Double Linked List

- Ilustrasi

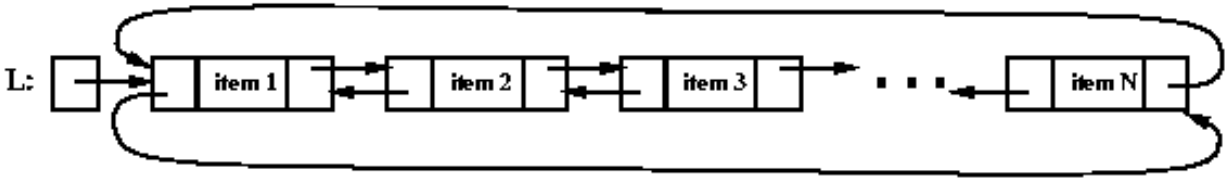


Circular Double Linked List

Circular, singly linked list:



Circular, doubly linked list:



Yang sudah dipelajari

- Konsep Manajemen Memori dalam program bahasa C
  - Penggunaan fungsi malloc(), calloc() dan free()
- Struktur Data Senarai Berantai (Linked List)
  - Single linked list
  - Double linked list

Soal 3

- Implementasikan double linked list untuk mencetak elemen list dari 1, 2, 3 … sampai 10. dan sebaliknya dari 10, 9, 8 sampai 1.

Referensi

- Robertson, Lesley Anne. (1992). *Students' guide to program design*. Oxford : Newnes
- Santner, Williams, and Notz (2003), Design and Analysis of Computer Experiments, Springer.
- Deitel & Deitel, *C How to Program*, Prentice Hall 1994 (2nd edition)
- Brookshear, J.G., *Computer Science: An Overview*, Benjamin-Cummings 2000 (6th edition)
- Kernighan & Ritchie, *The C Programming Language*, Prentice Hall