Praktikum
ALOKASI MEMORI DINAMIS

A. TUJUAN

1. Menjelaskan tentang cara menentukan ukuran objek
2. Menjelaskan tentang cara mengalokasikan memori
3. Menjelaskan penggunaan fungsi free() untuk membebaskan kembali memory
4. Menjelaskan tentang cara mengalokasikan memory untuk objek struct dengan jumlah dan data yang diinputkan

B. PERCOBAAN

1. Menggunakan fungsi sizeof() untuk menentukan ukuran objek

/* File program : size.c

```
Menggunakan fungsi sizeof() untuk menentukan ukuran objek */

#include <stdio.h>

typedef struct employee_st {
    char name[40];
    int id;
} Employee;

main(){
    int myInt;
    Employee john;

    printf("Size of int is %d\n",sizeof(myInt));
        //The argument of sizeof is an object

    printf("Size of int is %d\n",sizeof(int));
        //The argument of sizeof is a data type

    printf("Size of Employee is %d\n",sizeof(Employee));
        //The argument of sizeof is an object

    printf("Size of john is %d\n",sizeof(john));
        //The argument of sizeof is a data type

    printf("Size of char is %d\n",sizeof(char));
    printf("Size of short is %d\n",sizeof(short));
    printf("Size of int is %d\n",sizeof(int));
    printf("Size of long is %d\n",sizeof(long));
    printf("Size of float is %d\n",sizeof(float));
    printf("Size of double is %d\n",sizeof(double));
}
```

2. Menggunakan fungsi malloc() untuk mengalokasikan memory

```
/* File program : alokasi1.c
```

```
Menggunakan fungsi malloc() untuk mengalokasikan memory */
#include <stdio.h>
#include <stdlib.h>
#include <string.h>
main(){
   char s1[] = "This is a sentence";
   char *pblok;

   pblok = (char *) malloc(strlen(s1) + 1);
   /* Remember that stings are terminated by the null
      terminator, "\0',  & the strlen returns the length of a
      string not including the terminator
   */

   if (pblok == NULL)
      printf("Error on malloc\n");

   else {
      strcpy(pblok,s1);
         printf("s1: %s\n", s1);
         printf("pblok: %s\n", pblok);
      }
 }
```

3. Menggunakan fungsi free() untuk membebaskan kembali memory

```
/* File program : alokasi2.c
Menggunakan fungsi free() untuk membebaskan kembali memory */

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

main(){
   char *pblok;

   pblok = (char *) malloc(500 * sizeof(char));

   if (pblok == NULL)
      puts("Error on malloc");
   else {
      puts("OK, alokasi memory sudah dilakukan");
      puts("------");
      free(pblok);
      pblok = NULL;
      puts("Blok memory telah dibebaskan kembali");
      puts(“dan pointernya sdh di-groundkan”);
   }
 }
```

4. Mengalokasikan memory untuk objek struct dengan jumlah dan data yang diinputkan kemudian menampilkannya

```
/* File program : employee.c
Mengalokasikan memory untuk objek struct dengan jumlah dan data
yang diinputkan kemudian menampilkannya */

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

typedef struct employee_st {
    char name[40];
    int id;
} Employee;

main()
{
    Employee *workers, *wpt;
    int num, i;

    printf("How many employees do you want ? ");
    scanf("%d",&num);

    workers = (Employee *) malloc(num * sizeof(Employee));
    if (workers == NULL)
    {
        printf("Unable to allocated space for employees\n");
        exit(0);
    }

    wpt = workers;
    for(i=1; i<=num; i++)
    {
        fflush(stdin);
        printf("Employee's name : ");
        gets(wpt->name);
        printf("Employee's id : ");
        scanf("%d", &wpt->id);
        wpt++;
    }
    puts("");
    wpt = workers;
    for(i=1; i<=num; i++)
    {
        printf("Employee %d is %s\n", wpt->id, wpt->name);
        wpt++;
    }

    free(workers);
    workers = NULL;
}
```

Bukalah modul 2point.ppt, kerjakan :

5. Q4
6. Q5
7. Q6 → gambar diagramnya