

Programmazione e Strutture Dati (PR&SD)

1° ANNO – Informatica
Prof. V. Fuccella

Pseudo-generics in C



Generics

- **Definizione:** strumento che permette la definizione di un tipo parametrizzato, che viene esplicitato successivamente in fase di compilazione (o linkaggio) secondo le necessità;
- Permettono di
 - Eseguire algoritmi su tipi di dati diversi
 - Applicare ADT su tipi di dati diversi
- Alcuni linguaggi supportano generics al compile time.
 - Esempio di *Java Collections*:
 - `List<String> words=new ArrayList<String>();`
 - `List<Integer> numbers=new ArrayList<Integer>();`

Bubble Sort

```
void swap_int(int *a, int *b){
    int temp = *a;
    *a = *b;
    *b = temp;
}
```

```
void bsort_int(int a[], int n)
{
    int i, j;
    for(i=1; i<n; i++)
        for(j=0; j<n-i; j++)
            if(a[j] > a[j+1])
                swap_int(&a[j], &a[j+1]);
}
```

```
void swap_string(char **a, char **b){
    char *temp = *a;
    *a = *b;
    *b = temp;
}
```

```
void bsort_string(char *a[], int n)
{
    int i, j;
    for(i=1; i<n; i++)
        for(j=0; j<n-i; j++)
            if(strcmp(a[j], a[j+1])>0)
                swap_string(&a[j], &a[j+1]);
}
```

Bubble Sort

```
void swap_int(int *a, int *b){
    int temp = *a;
    *a = *b;
    *b = temp;
}
```

```
void bsort_int(int a[], int n)
{
    int i, j;
    for(i=1; i<n; i++)
        for(j=0; j<n-i; j++)
            if(a[j] > a[j+1])
                swap_int(&a[j], &a[j+1]);
}
```

```
void swap_string(char **a, char **b){
    char *temp = *a;
    *a = *b;
    *b = temp;
}
```

```
void bsort_string(char *a[], int n)
{
    int i, j;
    for(i=1; i<n; i++)
        for(j=0; j<n-i; j++)
            if(strcmp(a[j], a[j+1])>0)
                swap_string(&a[j], &a[j+1]);
}
```

Main (I/O)

```
int main(){
    int i, n = 5;
    int arr[n];
    printf("Introduci il vettore: ");
    for(i=0; i<n; i++){
        scanf("%d",&arr[i]);
    }
    bsort_int(arr, n);
    printf("Vettore ordinato: ");
    for(i=0; i<n; i++){
        printf("%d ",arr[i]);
    }
}
```

```
int main(){
    int i, n = 5;
    char *arr[5];
    printf("Introduci il vettore: ");
    for(i=0; i<n; i++){
        arr[i] = malloc(20*sizeof(char));
        scanf("%s",arr[i]);
    }
    bsort_string(arr, n);
    printf("Vettore ordinato: ");
    for(i=0; i<n; i++){
        printf("%s ",arr[i]);
    }
}
```

Main (I/O)

```
int main(){
    int i, n = 5;
    int arr[n];
    printf("Introduci il vettore: ");
    for(i=0; i<n; i++){
        scanf("%d",&arr[i]);
    }
    bsort_int(arr, n);
    printf("Vettore ordinato: ");
    for(i=0; i<n; i++){
        printf("%d ",arr[i]);
    }
}
```

```
int main(){
    int i, n = 5;
    char *arr[5];
    printf("Introduci il vettore: ");
    for(i=0; i<n; i++){
        arr[i] = malloc(20*sizeof(char));
        scanf("%s",arr[i]);
    }
    bsort_string(arr, n);
    printf("Vettore ordinato: ");
    for(i=0; i<n; i++){
        printf("%s ",arr[i]);
    }
}
```




Obiettivo

- Implementare algoritmi e ADT che siano in grado di funzionare, di volta in volta, con il tipo di dati desiderato
- Esempio: problema dell'ordinamento
 1. Interi: ordinare una sequenza di numeri
 2. Stringhe: mettere un elenco di nomi in ordine alfabetico
 3. Dati strutturati: ordinare i record di studenti secondo la matricola



Soluzione

- Generalizzare il problema dell'ordinamento in modo che possa funzionare con i 3 tipi specificati in precedenza; interi, stringhe e strutture
- Come procedere:
 - Creare un tipo generico *Item* (interfaccia) che supporti input, output e confronto
 - Modificare le librerie in modo che operino sul tipo *Item*
 - Realizzare *Item* in 3 file .c che supportano le 3 varianti intero, stringa e struttura (es.: *Studente*)
 - Linkare ed eseguire separatamente (con *make*) le 3 varianti



Supporto del C

Puntatori void*

- È possibile dichiarare un puntatore ad un tipo non specificato

- `void *p_void;`

- E poi assegnarlo al tipo desiderato

- `int* p_int = p_void;`

- `char* p_char = p_void;`

- ```
struct studente{
 char nome[20];
 int matricola;
};
```

```
typedef struct studente *Studente;
Studente p_studente = p_void;
```