

1.4 Quelques réflexions sur la complexité à partir d'exemples

Parcours

```
1. for (i=1 ; i<length(t) ; i++)
2.   print t[i]
```

```
1. for (i=1 ; i<n ; i++)
2.   read(a) ;
3.   b = a*9/5 + 32 ;
4.   print b ;
```

Notions : – taille de l'input
 – *random-access machine*
 – temps d'exécution – fonction linéaire *vs.* quadratique

Recherche (linéaire)

```
1. for (i=1 ; i<length(t) ; i++)
2.   if (t[i]==key) print(i) ;
```

```
1. i= 1
2. while(t[i] <> key)
3.   i = i + 1 ;
4. if (t[i]==key) print i ;
```

Notions : – cas le pire et cas moyen
 – taux (ordre) de croissance

Tri

champion:

```
1. best = t[1] ;
2. for (i=2 ; i<length(t) ; i++)
3.   if (best < t[i]) best = t[i]
```

champion(d,f) :

```
1. best = t[d] ;
2. for (i=d+1 ; i<f ; i++)
3.   if (best < t[i]) best = t[i]
4. return best
```

tri :

```
1. for (i=1 ; i<length(t) ; i++)
2.   t[i] = champion(i,length(t)) ;
```