Algoritmos e Estruturas de Dados I

Prof. Lúcio Mauro Pereira

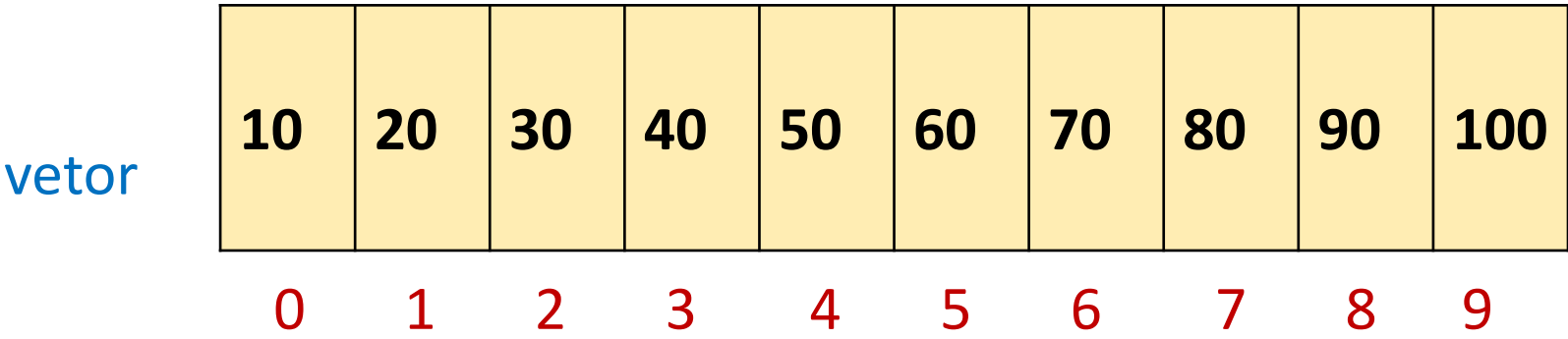
16/10/2024

Aulas anteriores

Tamanho definido pela coleção

Vetor como coleção de dados

```
int main()
{
    int vetor[] = { 10,20,30,40,50,60,70,80,90,100 };
    escrevaArray( vetor );
    return 0;
}// fim main()
```

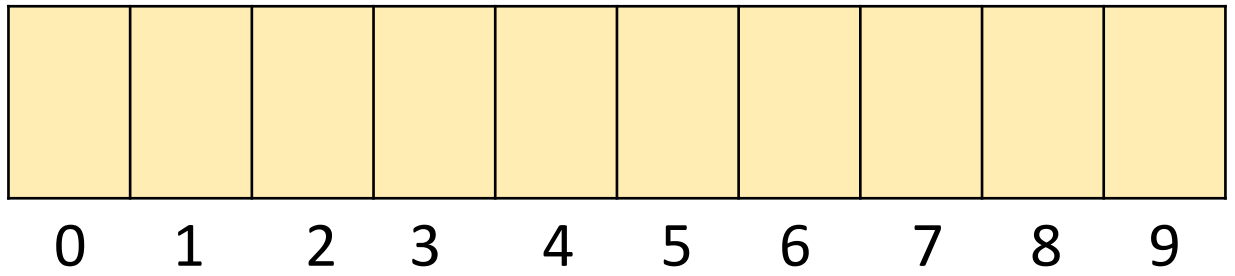


Tamanho definido ao declarar o vetor

Ler idade da turma (10 alunos).
Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    for( int i=0 ; i<10 ; i++ )
    {
        printf("\nDigite a %i.a idade: ", i+1 );
        scanf ( "%i", &idades[i] );
    }//fim for(i)
} //fim leiaIdades()
```

```
int main()
{
    int idades[10];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima",qtAcimaArray(idades,media) );
    return 0;
}// fim main()
```



Tamanho definido em constante global

Ler idade da turma (10 alunos).
Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    for( int i=0 ; i < MAX ; i++ )
    {
        printf("\nDigite a %i.a idade: ", i+1 );
        scanf ( "%i", &idades[i] );
    }//fim for(i)
}//fim leiaIdades()
```

```
const int MAX = 10;  //Constante global

int main()
{
    int idades[ MAX ];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima", qtAcimaArray(idades,media) );
    return 0;
}// fim main()
```

Tamanho definido em #define

Ler idade da turma (10 alunos).
Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    for( int i=0 ; i< _MAX ; i++ )
    {
        printf("\nDigite a %i.a idade: ", i+1 );
        scanf ( "%i", &idades[i] );
    }//fim for(i)
}//fim leiaIdades()
```

```
#define _MAX 10 //Define tamanho físico do vetor

int main()
{
    int idades[ _MAX ];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima", qtAcimaArray(idades,media) );
return 0;
}// fim main()
```

Tamanho definido em valor lido

Ler idade de uma turma com *n* alunos, sendo *n* um valor lido

```
int MAX ; // Variável global

int main()
{
    MAX = ???;
    int idades[ MAX ];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima", qtAcimaArray(idades,media) );
return 0;
}// fim main()
```

Ler idade de uma turma com *n* alunos, sendo *n* um valor lido

```
int leiaTamanho()
{
    int N;

    do {
        printf("\nNúmero de itens: " );
        scanf ( "%i", &N );


    }while( N<1 );
    return N;
}//fim leiaTamanho()
```

```
int MAX ; // Variável global

int main()
{
    MAX = leiaTamanho();
    int idades[ MAX ];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima", qtAcimaArray(idades,media) );
return 0;
}// fim main()
```

Ler idade de uma turma com *n* alunos, sendo *n* um valor lido

```
int leiaTamanho()
{
    int N;
    bool ERRO;
    do {
        printf("\nNúmero de itens: " );
        scanf ( "%i", &N );


    }while( ERRO );
    return N;
}//fim leiaTamanho()
```

```
int MAX ; // Variável global

int main()
{
    MAX = leiaTamanho();
    int idades[ MAX ];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima", qtAcimaArray(idades,media) );
return 0;
}// fim main()
```

Ler idade de uma turma com *n* alunos, sendo *n* um valor lido

```
int leiaTamanho()
{
    int N;
    bool ERRO;
    do {
        printf("\nNúmero de itens: " );
        scanf ( "%i", &N );
        ERRO = N<1;

    }while( ERRO );
    return N;
}//fim leiaTamanho()
```

```
int MAX ; // Variável global

int main()
{
    MAX = leiaTamanho();
    int idades[ MAX ];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima", qtAcimaArray(idades,media)
);    return 0;
}// fim main()
```

Ler idade de uma turma com *n* alunos, sendo *n* um valor lido

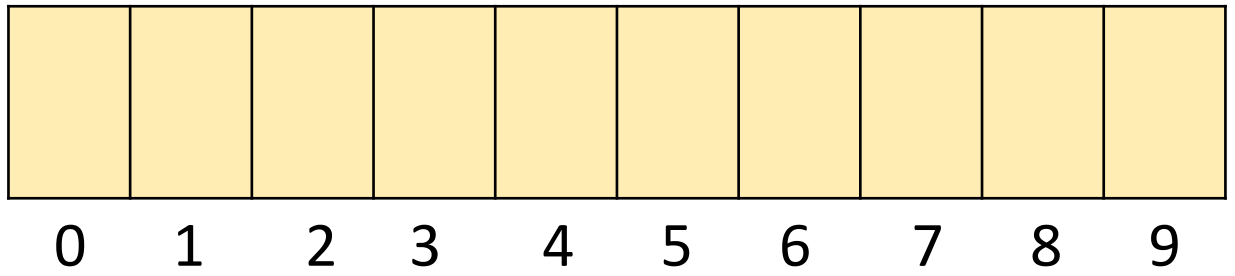
```
int leiaTamanho()
{
    int N;
    bool ERRO;
    do {
        printf("\nNúmero de itens: " );
        scanf ( "%i", &N );
        ERRO = N<1;
        if( ERRO ) printf("\nErro!");
    }while( ERRO );
    return N;
}//fim leiaTamanho()
```

```
int MAX ; // Variável global

int main()
{
    MAX = leiaTamanho();
    int idades[ MAX ];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima", qtAcimaArray(idades,media) );
return 0;
}// fim main()
```

Tamanho estático, interrupção com *flag*

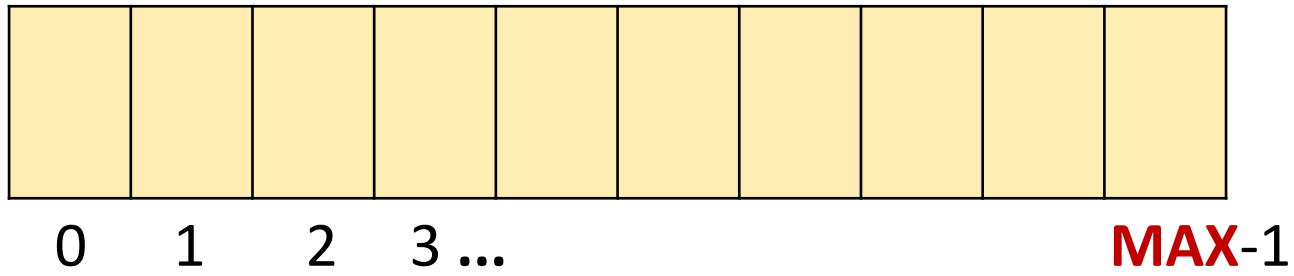
Ler idade da turma. *Flag:* idade igual a *0.* Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    int i=0;
    printf("\nDigite a 1.a idade: " );
    scanf ( "%i", &idades[0] );
    while( i< 9 && idades[i]!=0)
    {
        i++;
        printf("\nDigite a %i.a idade: ", i+1 );
        scanf ( "%i", &idades[i] );
    }//fim while()
}//fim leiaIdades()
```



Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

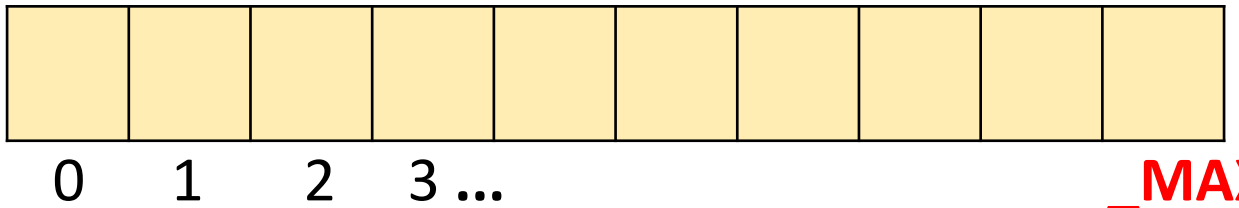
```
void leiaIdades(int idades[])
{
    int i=0;
    printf("\nDigite a 1.a idade: " );
    scanf ( "%i", &idades[0] );
    while( i< MAX-1 && idades[i]!=0)
    {
        i++;
        printf("\nDigite a %i.a idade: ", i+1 );
        scanf ( "%i", &idades[i] );
    }//fim while()
}//fim leiaIdades()
```



Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    int i=0;
    printf("\nDigite a 1.a idade: " );
    scanf ( "%i", &idades[0] );
    while( i< _MAX-1 && idades[i]!=0)
    {
        i++;
        printf("\nDigite a %i.a idade: ", i+1 );
        scanf ( "%i", &idades[i] );
    }//fim while()
}//fim leiaIdades()
```

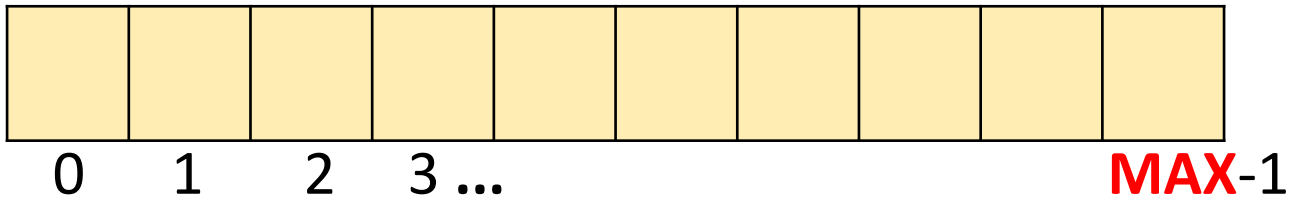
```
#define _MAX 10

int main()
{
    int idades[ _MAX ];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima", qtAcimaArray(idades,media) );
return 0;
}// fim main()
```



Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    int i=0;
    printf("\nDigite a 1.a idade: " );
    scanf ( "%i", &idades[0] );
    while( i< MAX-1 && idades[i]!=0)
    {
        i++;
        printf("\nDigite a %i.a idade: ", i+1 );
        scanf ( "%i", &idades[i] );
    }//fim while()
}//fim leiaIdades()
```

```
const int MAX = 10;

int main()
{
    int idades[ MAX ];
    leiaIdades( idades );
    float media = mediaArray( idades );
    printf("\nMedia = %f", media );
    print("\n%i acima", qtAcimaArray(idades,media) );
return 0;
}// fim main()
```

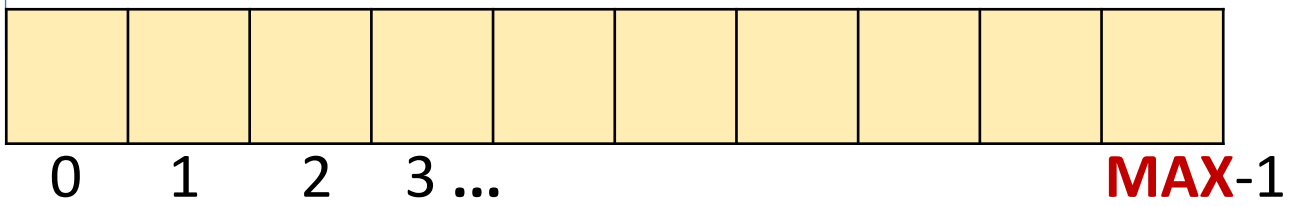


Tamanho físico *versus* tamanho lógico

Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    int i=0;
    printf("\nDigite a 1.a idade: " );
    scanf ( "%i", &idades[0] );
    while( i< MAX-1 && idades[i]!=0)
    {
        i++;
        printf("\nDigite a %i.a idade: ", i+1 );
        scanf ( "%i", &idades[i] );
    }//fim while()

}//fim leiaIdades()
```

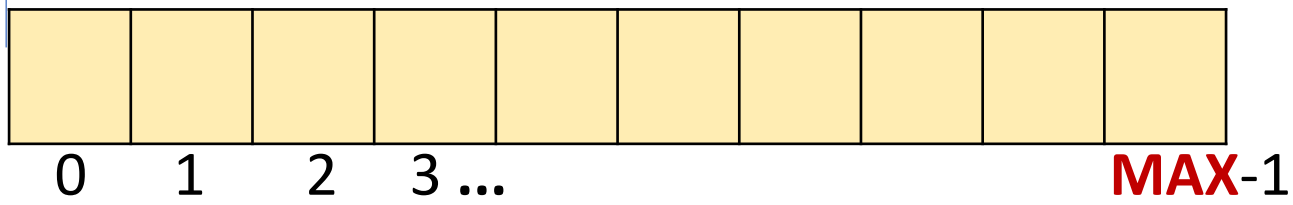
```
const int MAX = 100;


int main()
{
    int idades[ MAX ];



    return 0;
}// fim main()
```



Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    int i= TAM;
    printf("\nDigite a %i.a idade: ", i+1 );
    scanf ( "%i", &idades[i] );
    while( i< MAX-1 && idades[i]!=0)
    {
        i++;
        printf("\nDigite a %i.a idade: ", i+1 );
        scanf ( "%i", &idades[i] );
    }//fim while()
    TAM = i;
} //fim leiaIdades()
```

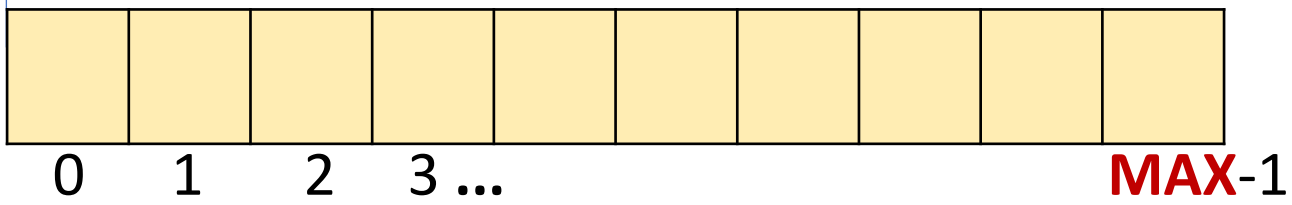
```
const int MAX = 100;
        int TAM =    0;

int main()
{
    int idades[ MAX ];




    return 0;
}// fim main()
```



Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{

    printf("\nDigite a %i.a idade: ", TAM +1 );
    scanf ( "%i", &idades[TAM] );
    while(TAM<MAX-1 && idades[TAM]!=0)
    {
        TAM ++;
        printf("\nDigite a %i.a idade: ", TAM+1 );
        scanf ( "%i", &idades[TAM] );
    }//fim while()

} //fim leiaIdades()
```

```
const int MAX = 100;
          int TAM =     0;

int main()
{
    int idades[ MAX ];



     return 0;
}// fim main()
```

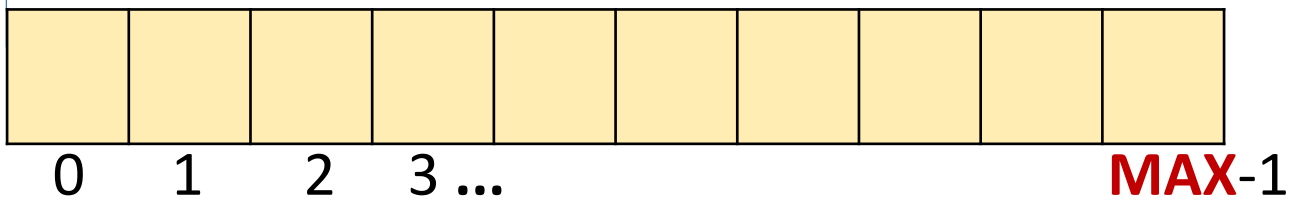


Novamente: funções atômicas

Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{

    printf("\nDigite a %i.a idade: ", TAM +1 );
    scanf ( "%i", &idades[TAM] );
    while(TAM<MAX-1 && idades[TAM]!=0)
    {
        TAM ++;
        printf("\nDigite a %i.a idade: ", TAM+1 );
        scanf ( "%i", &idades[TAM] );
    }//fim while()

} //fim leiaIdades()
```

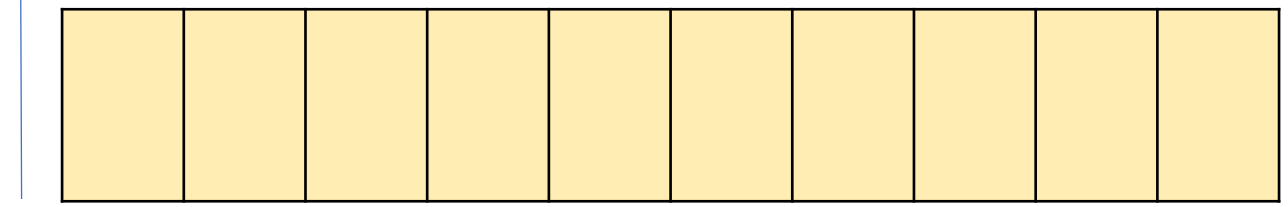
```
const int MAX = 100;
        int TAM =     0;

int main()
{
    int idades[ MAX ];



     return 0;
}// fim main()
```



Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    leiaIdade( idades );
    while( TAM < MAX && idades[TAM-1]!=0)
    {
        leiaIdade( idades );
    }//fim while()
}//fim leiaIdades()
```

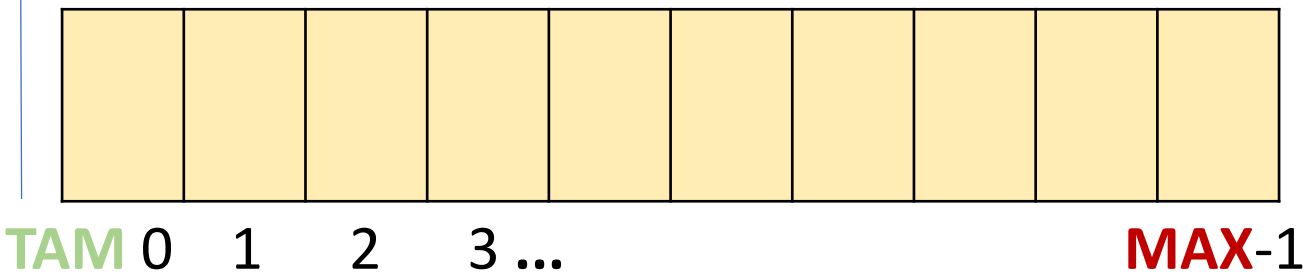
```
const int MAX = 100;
      int TAM =     0;

int main()
{
    int idades[ MAX ];


    return 0;
}// fim main()
```



Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{

    do
    {
        leiaIdade( idades );
    } while( TAM < MAX && idades[TAM-1]!=0)
}//fim leiaIdades()
```

```
const int MAX = 100;
      int TAM =     0;

int main()
{
    int idades[ MAX ];



    return 0;
}// fim main()
```

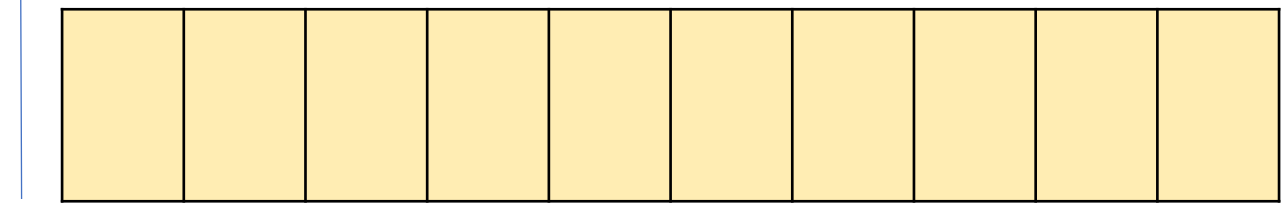


Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    do{
        leiaIdade( idades );
    }while( TAM < MAX && idades[TAM-1]!=0);
}//fim leiaIdades()
```

```
void leiaIdade(int idades[])
{
    printf("\nDigite a %i.a idade: ", TAM+1);
    scanf ( "%i", &idades[TAM] );
    TAM++;
}//fim leiaIdade()
```

```
const int MAX = 100;
      int TAM =   0;

int main()
{
    int idades[ MAX ];


    return 0;
}// fim main()
```

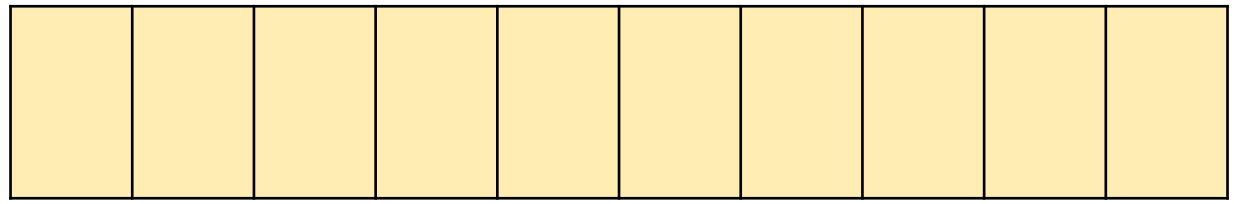


Ler idade da turma. *Flag:* idade igual a *0*. Identificar o número de idades acima da média.

```
void leiaIdades(int idades[])
{
    do{
        leiaIdade( idades );
    }while( TAM < MAX && idades[TAM-1]!=0)
}//fim leiaIdades()
```

```
void leiaIdade(int idades[])
{
     if(TAM < MAX) {
        printf("\nDigite a %i.a idade: ", TAM+1);
        scanf ( "%i", &idades[TAM] );
        TAM++;
     }
}//fim leiaIdade()
```

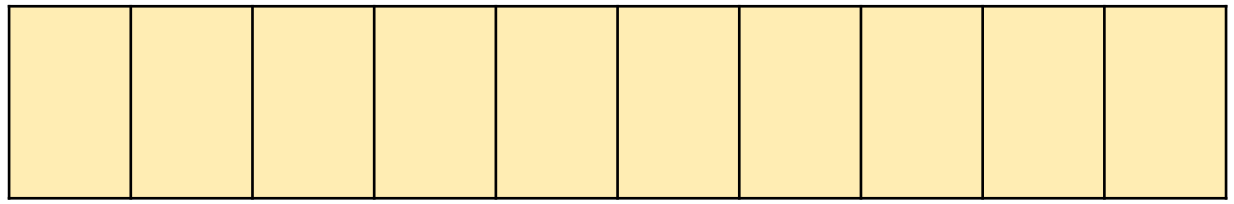
```
const int MAX = 100;
          int TAM =     0;

int main()
{
    int idades[ MAX ];


     return 0;
}// fim main()
```



Exemplo: Manipulação do arranjo gerenciada por menu

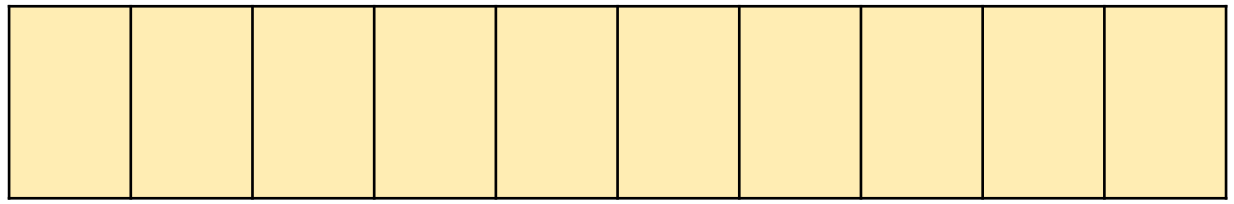
```
const int MAX = 100;
      int TAM =     0;

int main()
{
    int idades[ MAX ];



    return 0;
}// fim main()
```

```
void leiaIdade(int idades[])
{  if(TAM < MAX) {


      printf("\n%i.a idade: ", TAM+1);
      scanf ("%i", &idades[TAM] );



    TAM++;
  }//fim if()
}//fim leiaIdade()
```



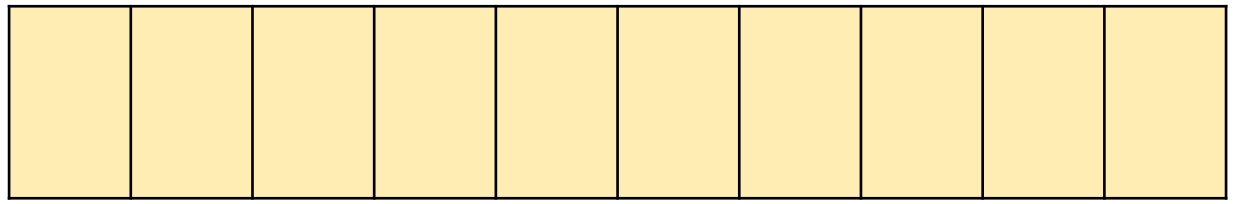
```
const int MAX = 100;
      int TAM =     0;

int main()
{
    int idades[ MAX ];



    return 0;
}// fim main()
```

```
void leiaIdade(int idades[])
{  if(TAM < MAX) {
     bool ERRO;

       printf("\n%i.a idade: ", TAM+1);
       scanf ("%i", &idades[TAM] );
       ERRO = idades[TAM] < 0;



     TAM++;
   }//fim if()
}//fim leiaIdade()
```



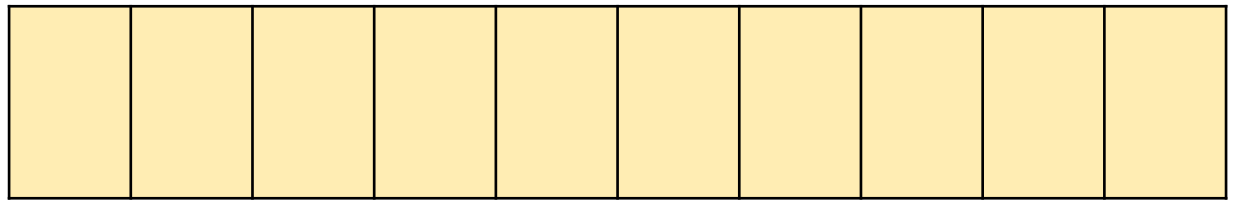
```
const int MAX = 100;
      int TAM =     0;

int main()
{
    int idades[ MAX ];



    return 0;
}// fim main()
```

```
void leiaIdade(int idades[])
{  if(TAM < MAX) {
     bool ERRO;

       printf("\n%i.a idade: ", TAM+1);
       scanf ("%i", &idades[TAM] );
       ERRO = idades[TAM] < 0;
       if (ERRO) printf("\nApenas >= 0");

     TAM++;
   }//fim if()
}//fim leiaIdade()
```



```
const int MAX = 100;
      int TAM =     0;

int main()
{
    int idades[ MAX ];



    return 0;
}// fim main()
```

```
void leiaIdade(int idades[])
{  if(TAM < MAX) {
     bool ERRO;
     do{
        printf("\n%i.a idade: ", TAM+1);
        scanf ("%i", &idades[TAM] );
        ERRO = idades[TAM] < 0;
        if (ERRO) printf("\nApenas >= 0");
      }while( ERRO );
      TAM++;
   }//fim if()
}//fim leiaIdade()
```



```
const int MAX = 100;
      int TAM =     0;

int main()
{
    int idades[ MAX ];
    int opcao;
    do{
          opcao=menu();
```

```
void leiaIdade(int idades[])
{  if(TAM < MAX) {
     bool ERRO;
     do{
        printf("\n%i.a idade: ", TAM+1);
        scanf ("%i", &idades[TAM] );
        ERRO = idades[TAM] < 0;
        if (ERRO) printf("\nApenas >= 0");
      }while( ERRO );
      TAM++;
   }//fim if()
}//fim leiaIdade()
```

```
    }while( opcao != 0 );
    return 0;
}// fim main()
```

```
int menu()
{    int OPCAO;
     bool ERRO;
     do{ printf("\n 0 – Sair");
         printf("\n 1 – Cadastrar");
         printf("\n 2 – Acima da média");
         printf("\n Sua opção: ");
         scanf ("%i", &OPCAO);
         ERRO = OPCAO<0 || OPCAO>2;
         if (ERRO) printf("\nOpção inválida");
     }while( ERRO );
     return OPCAO;
}//fim menu()


void leiaIdade(int idades[])
{  if(TAM < MAX) {
     bool ERRO;
     do{
        printf("\n%i.a idade: ", TAM+1);
        scanf ("%i", &idades[TAM] );
        ERRO = idades[TAM] < 0;
        if (ERRO) printf("\nApenas >= 0");
      }while( ERRO );
      TAM++;
   }//fim if()
}//fim leiaIdade()
```

```
const int MAX = 100;
        int TAM =     0;

int main()
{
    int idades[ MAX ];
    int opcao;
    do{
            opcao=menu();









     }while( opcao != 0 );
     return 0;
}// fim main()
```

```
int menu()
{    int OPCAO;
     bool ERRO;
     do{ printf("\n 0 – Sair");
         printf("\n 1 – Cadastrar");
         printf("\n 2 – Acima da média");
         printf("\n Sua opção: ");
         scanf ("%i", &OPCAO);
         ERRO = OPCAO<0 || OPCAO>2;
         if (ERRO) printf("\nOpção inválida");
     }while( ERRO );
     return OPCAO;
}//fim menu()


void leiaIdade(int idades[])
{  if(TAM < MAX) {
     bool ERRO;
     do{
        printf("\n%i.a idade: ", TAM+1);
        scanf ("%i", &idades[TAM] );
        ERRO = idades[TAM] < 0;
        if (ERRO) printf("\nApenas >= 0");
      }while( ERRO );
      TAM++;
   }//fim if()
}//fim leiaIdade()
```

```
const int MAX = 100;
      int TAM =     0;

int main()
{
    int idades[ MAX ];
    int opcao;
    do{
          opcao=menu();
          switch(opcao){
            case 1 :
                    break;
            case 2 :



                    break;
            default: printf("\nOpção inválida\n");
          }//fim switch(opcao)
     }while( opcao != 0 );
     return 0;
}// fim main()
```

```c
int menu()
{    int OPCAO;
     bool ERRO;
     do{ printf("\n 0 – Sair");
         printf("\n 1 – Cadastrar");
         printf("\n 2 – Acima da média");
         printf("\n Sua opção: ");
         scanf ("%i", &OPCAO);
         ERRO = OPCAO<0 || OPCAO>2;
         if (ERRO) printf("\nOpção inválida");
     }while( ERRO );
     return OPCAO;
}//fim menu()

void leiaIdade(int idades[])
{  if(TAM < MAX) {
      bool ERRO;
      do{
         printf("\n%i.a idade: ", TAM+1);
         scanf ("%i", &idades[TAM] );
         ERRO = idades[TAM] < 0;
         if (ERRO) printf("\nApenas >= 0");
       }while( ERRO );
       TAM++;
    }//fim if()
}//fim leiaIdade()
```

```c
const int MAX = 100;
          int TAM =     0;

int main()
{
     int idades[ MAX ];
     int opcao;
     do{
               opcao=menu();
               switch(opcao){
                  case 1 : leiaIdade(idades);
                           break;
                  case 2 :



                           break;
                  default: printf("\nOpção inválida\n");
               }//fim switch(opcao)
      }while( opcao != 0 );
      return 0;
}// fim main()
```

```c
int menu()
{    int OPCAO;
     bool ERRO;
     do{ printf("\n 0 – Sair");
         printf("\n 1 – Cadastrar");
         printf("\n 2 – Acima da média");
         printf("\n Sua opção: ");
         scanf ("%i", &OPCAO);
         ERRO = OPCAO<0 || OPCAO>2;
         if (ERRO) printf("\nOpção inválida");
     }while( ERRO );
     return OPCAO;
}//fim menu()


void leiaIdade(int idades[])
{  if(TAM < MAX) {
     bool ERRO;
     do{
        printf("\n%i.a idade: ", TAM+1);
        scanf ("%i", &idades[TAM] );
        ERRO = idades[TAM] < 0;
        if (ERRO) printf("\nApenas >= 0");
      }while( ERRO );
      TAM++;
   }//fim if()
}//fim leiaIdade()
```

```c
const int MAX = 100;
          int TAM =     0;

int main()
{
     int idades[ MAX ];
     int opcao;
     do{
             opcao=menu();
             switch(opcao){
               case 1 : leiaIdade(idades);
                        break;
               case 2 : float media=mediaArray(idades);
                        printf("\nHá %i acima media",
                           qtAcimaArray(idades,media) );
                        break;
               default: printf("\nOpção inválida\n");
             }//fim switch(opcao)
      }while( opcao != 0 );
      return 0;
}// fim main()
```

Alguns desafios postos

Lista 23 – Exercício 1

Construa uma função que receba um arranjo de reais e o seu tamanho. A função deverá trocar o primeiro elemento do arranjo com o último.

Lista 23 – Exercício 2

Construa uma função que receba um arranjo de reais e o seu tamanho. A função deverá trocar dois elementos de lugar. As duas posições deverão também ser parametrizadas.

Lista 23 – Exercício 3

Construa uma função que receba um arranjo de reais e desloque o **maior** valor do arranjo para a **última** posição.

Lista 23 – Exercício 5

Construa uma função que receba um arranjo de reais e ordene-o em ordem crescente.

Lista 23 – Exercício 6

Construa uma função que receba um valor inteiro relativo a um mês do ano [1..12]. A função deverá retornar o número de dias que o mês possui.