SPM 2102 PROGRAMMING LANGUAGE 1

ARRAY

By

NORAH MD NOOR

ARRAYS

This topic will cover

- Declaration of one dimensional array
- Setting the initial value in a one dimensional array
- Array and function
- Strings

ARRAYS	WITHOUT ARRAYS

char gred[4]={'A','B', 'C'} ;	char gred1= "A";
	char gred2= "B";
	char gred3= "C";

int nombor[] = {10,20,30,40,} ;	int nombor1 = 10 ; int nombor2 = 20 ; int nombor3 = 30 ; int nombor4 = 40 ;

ARRAYS

- A data structure that can be used to hold a list of values from related variables

- The most suitable technique to display or read collective data (more than one)

- As an example: the marks for a list of a hundred students can be displayed by using only one variable

Eg : Marks for a list of a hundred students

```
int markah[]= {88,81,83,86,89,82);
```

Rather than

```
int markah1 = 88;
int markah2 = 81;
int markah3 = 83;
int markah4 = 86;
int markah5 = 89;
int markah6 = 82;
```

ARRAYS

- Array in C++ refers to a collection of values that:
  - has the same type of data

Eg :

int numb [] = {1,2,3,4,5};

char huruf [2] =’, {‘a’,’b’,’c’,d’}

ARRAYS

- Array in C++ refers to a collection of values that:
  – uses a common or single name

Eg :

```
int numb [] = {1,2,3,4,5};
char huruf [2] =’, {‘a’,’b’,’c’,d’}
```

ARRAYS

- Array in C++ refers to a collection of values that:
  - has specified size
  - has subscript/index
    - Subscript is the memory location on the computer for a data.
    - Subscripts starts with the value 0.

ONE DIMENSIONAL ARRAY

ONE DIMENSIONAL ARRAY

- A one dimensional array can be declared as follows:

var_type var_name[size]

Variable_type	Variable_name	Size
int	huruf	4
int huruf [4]		

- var_type is basic data type or base type (char, int, float, double)
- variable name as usual
- size defines how many elements the array will hold

ONE DIMENSIONAL ARRAY

- **The comparison of using variables and array :**

1. variables

int mark1 = 80;

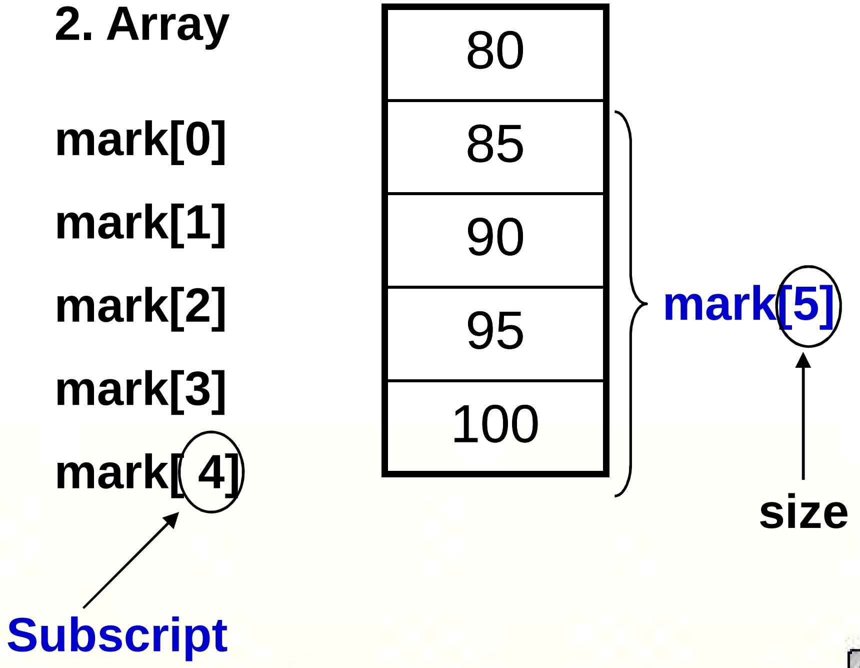
int mark2 = 85;

int mark3 = 90;

int mark4 = 95;

int mark5 = 100;

2. Array



mark[0]

mark[1]

mark[2]

mark[3]

mark[4]

80

85

90

95

100

mark[5]

size

Subscript

ONE DIMENSIONAL ARRAY

Example :

```
#include <iostream.h>
#include <conio.h>
void main ()
{
int nombor[4] = {1,2,3,4} ; //size
cout<<nombor[2] ;
getch();
}
```

PROCESSING ARRAYS

PROCESSING ARRAYS

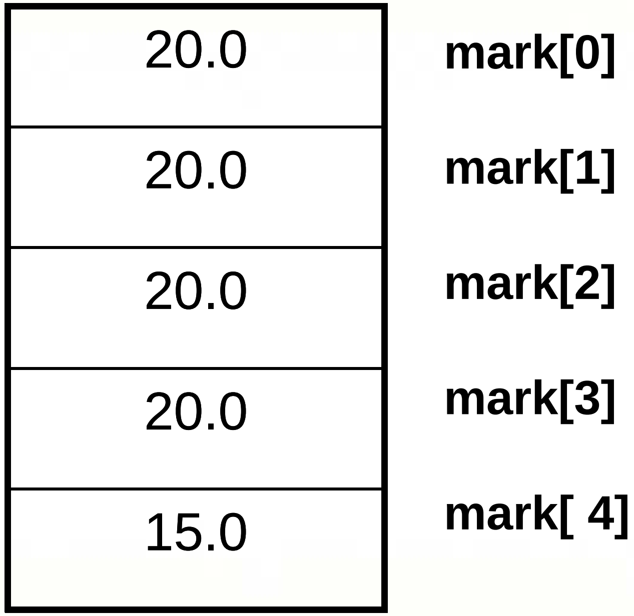
- An index identifies a specific position of an element within an array
- In C++, all arrays have 0 as the index of their first element
- Example : int sample [10]
- Because sample has ten elements, it has index values of 0 through 9.

PROCESSING ARRAYS

Example for storing exam marks of 5 students

```
float mark[5];
for(int i=0; i < 4; i++)
    mark[i] = 20.0;
mark[4] = 15.0;
```

variable type index



20.0	mark[0]
20.0	mark[1]
20.0	mark[2]
20.0	mark[3]
15.0	mark[4]

```
#include<iostream.h>
#include<conio.h>
void main()
{
const int size = 4;
int arrays[size];
for (int value = 0 ; value < size ; value++)
{
arrays[value]=0;
cout<<" Bil. Array "<<value<<" = "<<arrays[value]<<"\n";
}
getch();
}
```

STRING TYPE ARRAY

- String → special arrays
- Char in data type and ending with null “\0”
- The variable of string can be declared as declared arrays eg : char variable_name[size]

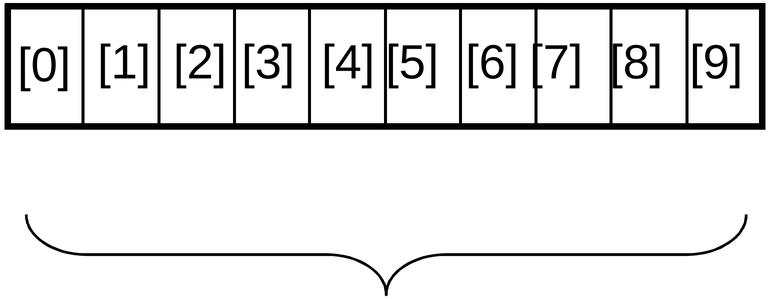
STRING TYPE ARRAY

- Example :
- char name[30] ;
- char alphabet[10] ;

- By this declaration, only 30 & 10 blocks can be memorized by this variable.

STRING TYPE ARRAY

- Eg :
- char name[10];



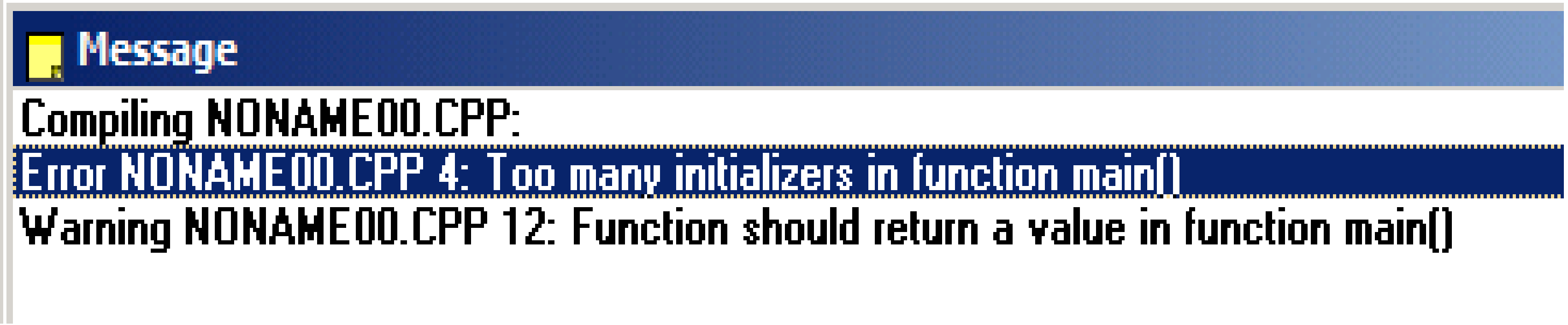
name

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]

10 sel ingatan

STRING TYPE ARRAY

```
#include<iostream.h>
int main()
{
char x[2] = {'a','b','c','d','e'}; // char x[2] = {“abcde”};
{
cout<<x[2];
}
}
```



```
Compiling NONAME00.CPP:
Error NONAME00.CPP 4: Too many initializers in function main()
Warning NONAME00.CPP 12: Function should return a value in function main()
```

STRING TYPE ARRAY

- Constant of string.
- Eg :
- char symbol [10] = {‘a’, ‘*’, ‘1’};

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
a	*	1							

STRING TYPE ARRAY

```
#include<iostream.h>
int main()
{
char x[5] = {'a','#','@','%','^'};
{
cout<<x[2];
}
}
```

[0]	[1]	[2]	[3]	[4]	[5]	[6]	[7]	[8]	[9]
a	#	@	%	^					

Output = @

STRING TYPE ARRAY

- String.h = String arrays modification
- Functions :
  - strcpy()
  - strcmp()
  - strlen()
  - strcat()

STRING TYPE ARRAY

- **strcpy()**
  - This function is used to copy a string to another string
  - Eg :

```
char sentence[10];
strcpy(sentence, “Hello”) ;
```

“Hello” string will be copied to string variable sentence

STRING TYPE ARRAY

- **strcmp(string1, string2)**
  - This strcmp will compare between two string in two variable
  - If string 1 == string 2, then 0 (false)
  - If string 1 > string 2, then positive (true)
  - If string 1 < string 2, then negative (true)

```
#include<iostream.h>
#include<string.h> //fungsi string
void main()
{
    char name_1[10], name_2[10];
    cout<<"\n Sila masukkan nama pertama : ";
    cin>>name_1;
    cout<<"\n Sila masukkan nama kedua : ";
    cin>>name_2;
    if ( strcmp ( name_1, name_2 ) )
            cout<<“Nama berbeza \n"; //defaultnya melihat
    perbezaan
    else
            cout<<“Nama sama \n";// else melihat persamaan
}
```

STRING TYPE ARRAY

- strlen() = Calculate length of string

```
#include<iostream.h>
#include<string.h>
void main()
    {
    char name_1[10] ;
    cout<<"\n Sila masukkan nama pertama: ";
    cin>>name_1;
    cout<<"\t" <<strlen(name_1);

    }
```

STRING TYPE ARRAY

strcat(string1, string2)

- This function is used to combine or append one string to another

- char str1[10] = {awal};
- strcat(str1, " akhir");

- str1 now will be "awal akhir".

That’s all ☺