--------------------------------------------------------------------

Q1: Is there any compile error with the following code (if any)?

unsigned int Arr[16];

3[Arr] = 7;

Explain: No compile error. During array indexing, the address of element is computed and the element is accessed as \*(Array+Index). And if we compute \*(Index+Array) it gives the same result. Hence in C & C++ both of these syntaxes are allowed.

--------------------------------------------------------------------

Q2: What is the difference between the following 3 statements?

const int \* px;

px is a variable pointer to an integer constant. ‘px' can be re-assigned.

int const \* px;

same as above, px is a variable pointer to an integer constant.

int \* const px;

px is a constant pointer to integer variable.

Is there any compile error for the following cases?

case1:

int x = 13;

const int \* px;

px = & x;

No Error. ‘px’ is not a constant, but a variable pointer to a integer constant. Hence it can be declared without being initialized. And px can be reassigned too.

case 2:

int x = 13;

int const \* px;

px = & x;

No Error. ‘px’ is not a constant, but a variable pointer to a integer constant. Hence it can be declared without being initialized. And px can be reassigned too.

case 3:

int x = 13;

int \* const px;

px = & x;

Error. ‘px’ is a constant pointer to an integer variable. Hence it **cannot** be declared without being initialized. Hence, the declaration yields error below:

error: uninitialized const 'px' [-fpermissive]

And px **cannot** be reassigned either since it is declared a constant and hence read only.

error: assignment of read-only variable 'px'

--------------------------------------------------------------------

Q3: Write a function to set or clear ith bit of a 32-bit register.

Where ith (0-based) := {0, 1, 2, …, 31 }

void reg\_set(volatile unsigned int \* pReg, int ith)

{

\*pReg |= (1<<ith);

}

void reg\_clear(volatile unsigned int \* pReg, int ith)

{

\*pReg &= ~(1<<ith);

}

--------------------------------------------------------------------

Q4: Write a swap function in C.

void swap(unsigned int \* px, unsigned int \*py)

{

unsigned int tmp;

tmp = \*px;

\*px = \*py;

\*py = tmp;

}

--------------------------------------------------------------------

Q5: What is the output of the following code? (Given: sizeof(unsigned int) is 4) Page 34

unsigned int Arr[16];

unsigned int a0 = (unsigned int) &Arr[0];

unsigned int a3 = (unsigned int) &Arr[3];

printf(“%d\n”, a3 – a0);

The result is 12.

a0 is the address of Arr[0] , or same as Arr’s address

a3 is the address of Arr[3], which is Arr + 3\*(sizeof(unsigned int)) = Arr + 3\*4 = Arr +12

Thus a3-a0 =( Arr+12) – Arr = 12

END