VauLSMorg

04 ADR-PTR - Addressing & Pointers
Aneka Soal Ujian Sistem Operasi
BinKadal, Sdn. Bhd. (editor)

1. **2016-2**

```
001 /* (c) 2016 Rahmat M. Samik-Ibrahim -- This is free software
005  * Assume (&ptrchr is 0x7FFFEEDDCCBB, order of bytes: little-endian) */
009 #define LINES  3
010 #include <stdio.h>
012 void printeq(int lines) {
013    while (lines-- > 0 ) printf("= = ");
014    printf("\n");
015 }
017 void main(void) {
018    int             ii;
019    unsigned char   dummy  = 'a';
020    unsigned char*  ptrchr = &dummy;
022    printeq(LINES);
023    printf(" dummy:  %c\n", dummy);
024    printf("*ptrchr: %c\n", *ptrchr);
025    printeq(LINES);
026    printf("%p\n", &ptrchr);
027    printeq(LINES);
028    ptrchr = (char*) &ptrchr;
029    for (ii=0; ii<6; ii++) {
030       printf("%X ", *ptrchr);
031       ptrchr++;
032    }
033    putchar('\n');
034    printeq(LINES);
035 }
```

(a) Write down the output of this program

2. **2017-1**

C Programing

```
001 /*
002  * (c) 2017 Rahmat M. Samik-Ibrahim
           -- This is free software
003  * REV00 Thu Mar 30 18:27:30 WIB 2017
004  * START Thu Mar 30 18:27:30 WIB 2017
005  * INT is 32 bit little endian
006  * 41H='A'; 42H='B'; 43H='C"; 44H='D'
007  */
008
009 #include <stdio.h>
010 char chrary[]="ZZZZ ZZZZ ";
```

```
011 void main(void) {
012    char  chrvar = 'M';
013    int   intvar = 0x41424344;
014    int*  intptr = (int*) chrary;
015    printf("YY.     chrary=%p\n",  chrary);
016    printf("ZZ.     intptr=%p\n",  intptr);
017    printf("01.     chrvar=%c\n",  chrvar);
018    printf("02.    *chrary=%c\n", *chrary);
019    printf("03. str chrary=%s\n",  chrary);
020    *intptr      =  intvar;
021    printf("04. str chrary=%s\n",  chrary);
022 }
```

Program Output (Line: 015, 016, 017, 018, 019, 021):

```
YY. chrary=0x600a08
```

3. **2017-2**

C Programing ADDR

```
001 /*
002  * (c) 2017 Rahmat M. Samik-Ibrahim
003  * http://rahmatm.samik-ibrahim.vlsm.org/
004  * This is free software.
005  * REV00 Mon Oct 16 21:15:03 WIB 2017
006  * START Mon Oct 16 21:15:03 WIB 2017
007  */
008
009 #include <stdio.h>
010
011 char* stringChar="HALLO";
012 char* stringPTR;
```

```
014 void main (void) {
015     stringPTR=stringChar;
016     printf ("ADDR1: %p VAL: %p STR: %s\n", &stringChar,
                                    stringChar, stringChar);
017     printf ("ADDR2: %p VAL: %p STR: %s\n", &stringPTR,
                                     stringPTR, stringPTR);
018     while (*(++stringPTR) != 0 ) {
019          printf ("ADDR3: %p VAL: %p STR: %s\n", &stringPTR,
                                          stringPTR, stringPTR);
020     }
021     printf ("End of String = %p\n", stringPTR);
022 }
```

Program Output:

```
ADDR1:  0x601038 VAL: 0x400674 STR: HALLO
ADDR2:  0x601048 VAL: 0x400674 STR: HALLO
```

4. **2018-1**

What is the output of this following program:

```
001 /* (c) 2018 This is a free program   */
002 /* Rahmat M. Samik-Ibrahim           */
003 /* The "array"   starts  at 0x601040 */
004 /* The "pointer" address is 0x601050 */
005
006 #include <stdio.h>
007
008 char  array[]="0123456789ABCDE";
009 char* pointer=array;
010 void  main(void) {
011    printf("START\n");
012    printf("%p\n", &pointer);
013    printf("%p\n",  pointer);
014    printf("%s\n",  pointer);
015    printf("%d\n",  pointer[15]);
016    printf("STOP\n");
017 }
```

```
018 /* Clue#1: All strings end with 0x00 */
019 /* Clue#2: Address=64 bit BIG ENDIAN */
020 /* Clue#3: ASCII '0' (Zero) is 0x30  */
021 /* Clue#4: ASCII 'A'        is 0x41  */
```

Initially, addresses 0x601040 - 0x60105F = 0x00. What will be in those addresses after executing the program?

Addresses (HEX)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000 0000 0060 1040																
0000 0000 0060 1050																

5. **2018-2 (44%)**

What is the output of this following program:

```
001 /* (c) 2018 This is a free program   */
002 /* Rahmat M. Samik-Ibrahim           */
003
004 #include <stdio.h>
005
006 unsigned char  string[]="0123456789ABCDE";
007 unsigned long  longst=(unsigned long) string;
008 unsigned char* achar;
009
010 void main(void) {
011    achar=(unsigned char*) &longst;
012    printf("(0) %p\n",  string);
013    printf("(1) %p\n", &longst);
014    printf("(2) %p\n",  &achar);
015    printf("(3) %p\n",   achar);
016    printf("(4) %#X\n", *achar);
017    printf("(5) %#16.16lx\n",longst);
018 }
```

```
019 /* This Clue #1 - Clue #5          */
020
021 /* 1: All strings end with 0x00    */
022 /* 2: Address=64 bit Little ENDIAN */
023 /* 3: ASCII '0' (Zero)  is 0x30    */
024 /* 4: ASCII 'A'         is 0x41    */
025 /* 5: Long Integer Size = 64bit    */
026

PROGRAM OUTPUT

(0) 0x601040
(1) 0x601050
(2) 0x601060
(3) 0x601050
(4) 0X40
(5) 0x0000000000601040
```

Initially, addresses 0x601040 - 0x60106F = 0x00. What will be in those addresses after executing the program?

Addresses (HEX)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000 0000 0060 1040																
0000 0000 0060 1050																
0000 0000 0060 1060																

6. **2019-1 (53.9%)**

```
Clue #1 - Clue #5:
1: All strings end with 0x00.
2: Address=64 bit Little ENDIAN.
3: ASCII 'A' is 0x41.
4: ASCII 'a' is 0x61.
5: Initially Addresses:
   0x0000 5566 7788 9900 - 991F = 0x00.

Program Output:
 &ch_array[0]=0x556677889910
  ch_array[0]=a

 &ch_pointer =0x556677889918
  ch_pointer =0x556677889909
 *ch_pointer =F
```

```
001 // (c) 2019 This is Free Software
002 // Rahmat M. Samik-Ibrahim 20190315-1352 R01
003
004 #include <stdio.h>
005 char  ch_array[]="abcdefg";
006 char* ch_pointer="ABCDEFG";
007


008 void main(void) {
009    printf(" &ch_array[0]=%p\n", &ch_array[0]);
010    printf("  ch_array[0]=%c\n",  ch_array[0]);
011    ch_pointer = ch_pointer + 5;
012    printf(" &ch_pointer =%p\n", &ch_pointer);
013    printf("  ch_pointer =%p\n",  ch_pointer);
014    printf(" *ch_pointer =%c\n", *ch_pointer);
015 }
```

What will be in these following addresses after executing the program (in **hexadecimal**)?

Addresses (HEX)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000 5566 7788 990X																
0000 5566 7788 991X																

7. **2019-2 (45%)**

```
001 // (c) 2019 This is Free Software
002 // Rahmat M. Samik-Ibrahim 20191021
003 /*
004 These are Clue #1 - Clue #5:
005 ============================
006 1:All strings end with 0x00.
007 2:A "string size" includes that 0x00.
008 3:All arrays start with index 0.
009 4:Address=64 bit Little ENDIAN.
010 5:ASCII '0' is 0x30.
011 The program output (lines 27-29):
012 ====================================
013 1. &string1[0]=0x556677889910
014 2. &string2[0]=0x556677889918
015 3. &stringPtr =0x556677889928
016 */
```

```
018 #include <stdio.h>
019 #include <string.h>
020 char  string1[]="0123456";
021 char  string2[]="0123456";
022 char* stringPtr;
023
024 void main(void) {
025    int   size1=sizeof(string1);
026    stringPtr=&string1[size1-1];
027    printf("1. &string1[0]=%p\n", &string1[0]);
028    printf("2. &string2[0]=%p\n", &string2[0]);
029    printf("3. &stringPtr =%p\n", &stringPtr);
030    printf("4.  stringPtr =%p\n",  stringPtr);
031    *stringPtr = '7';
032    printf("5.  STRING:   %s\n", &string1[0]);
033 }
```

(a) Program Output (line 30) (46%): _____ _____

(b) Program Output (line 32) (31%): _____ _____

(c) What will be in these following addresses after executing the program (in **hexadecimal**) (49%)?

Addresses (HEX)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000 5566 7788 991X																
0000 5566 7788 992X																

8. **2020-1**

```
001 // (c) 2020 This is Free Software
002 // Rahmat M. Samik-Ibrahim 2020
003 // R03 0310Tue1715
004 /*
005 This Clue #1 - Clue #5:
006 ========================
007 1: All strings end with 0x00.
008 2: All arrays start with index 0.
009 3: Address=64 bit Little ENDIAN.
010 4: ASCII '0' is 0x30.
011 5: ASCII 'A' is 0x41.
012 The first 3 lines of program output:
013 ====================================
014 1. 0X0000556677665520
015 2. 0X0000556677889918
016 3. 0X0000556677889910
017 */
```

```
019 #include <stdio.h>
020 #include <string.h>
021 typedef unsigned long UL;
022 char* stringptr="0123456";
023 char  string1[]="89ABCDE";
024
025 void main(void) {
026     printf("1. %#16.16lX\n", (UL)  stringptr);
027     printf("2. %#16.16lX\n", (UL) &stringptr);
028     printf("3. %#16.16lX\n", (UL) &string1[0]);
029
030     printf("4. %#16.16lX\n", (UL) &string1[6]);
031     printf("5. %#X %c\n",string1[6], string1[6]);
032     printf("6. %#X %c\n",*stringptr, *stringptr);
033     stringptr++;
034     printf("7. %#16.16lX\n", (UL)  stringptr);
035     printf("8. %#X %c\n",*stringptr, *stringptr);
036 }
```

Program Output:

(a) (line 30) _____ _____ _____

(b) (line 31) _____ _____ _____

(c) (line 32) _____ _____ _____

(d) (line 34) _____ _____ _____

(e) (line 35) _____ _____ _____

(f) **INITIALLY**, addresses 0x5566 7766 5520 - 0x5566 7766 552F, and 0x5566 7788 9910 - 0x5566 7788 991F = 0; What will be in those addresses after executing the program (in **hexadecimal**)?

Addresses (HEX)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000 5566 7766 552X																
0000 5566 7788 991X																

9. **2022-2 (47.3%)**

Program "`mymemory3.c`" a shortened version of the "`mymemory2.c`" program in the last **WEEK 05** assignment. See next page for output program "`mymemory3`" and source code "`mymemory3.c`" (Line numbers are added).

(a) (45%) Based on the output of the "`mymemory3`" program, what is the "**Total usable main memory size**" of that system? Answer:

Total Memory: _____ MB.

(b) (70%) Based on the output of the "`mymemory3`" program, what is the "**Total swap space size**" of that system? What was initially the "**free swap space still available size**"? Answer:

Total Swap Size: _____ _____ MB.

Free Available Size: _____ MB.

(c) (31%) What is the name of the Linux system call that returns specific statistics on memory and swap usage above?

System Call Name: _____

(d) (36%) Function `malloc()` is a C library routine that allocates memory dynamically. What happens to the memory when the malloc() function is called in line 34 of the program? What happens to the memory when the allocated memory is used in line 38? Explain and show any lines in the program or output lines of the program that support your explanation!

Explain lines 34 and 38 (Max. 4 lines!):

```
#### OUTPUT + LINES of mymemory3    ###############

01 ZCZC RAM         71 MB
02 ZCZC FREE        26 MB
03 ZCZC BUFFER       1 MB
04 ZCZC SWAP       975 MB
05 ZCZC FREESW     940 MB
06 ZCZC FREE1       26 MB
07 ZCZC FREESW1    940 MB
08 ZCZC FREE2        7 MB
09 ZCZC FREESW2    442 MB
10 ZCZC ADDR 01 0X000055E3A0079155 printMyAddress()
11 ZCZC ADDR 02 0X000055E3A0079192 main()
12 ZCZC ADDR 03 0X000055E3A007C040 &pcounter
13 ZCZC ADDR 04 0X00007FCC6D324010 intArray
14 ZCZC ADDR 05 0X00007FCC8D378CF0 printf()
15 ZCZC ADDR 06 0X00007FFFCEC2F260 &intArray
16 ZCZC ADDR 07 0X00007FFFCEC2F26C &localdummy
11 ZCZC ADDR 08 0X00007FFFCEC2F270 &guestInfo

### mymemory3.c ###################################

001 // Copyright (C) 2022 C. BinKadal
002 // This program is free script/software.
003 // REV01: Tue 11 Oct 2022 19:00
004 // START: Tue 11 Oct 2022 18:00
005
006 #include <stdio.h>
007 #include <stdlib.h>
008 #include <unistd.h>
009 #include <sys/sysinfo.h>
010
011 typedef char*           String;
012 typedef int*            IntPtr;
013 typedef unsigned long   UL;
014 typedef void*           AnyAddrPtr;
015 typedef struct sysinfo  SYSINFO;
017 int pcounter=1;
018 void printMyAddress (AnyAddrPtr address, String message) {
019     printf("ZCZC ADDR %2.2d %#16.16lX %s\n",
020            pcounter++, (UL) address, message);
021 }


023 #define ArraySize   128*1024*1024
024 int main(void) {
025     SYSINFO guestInfo;
026     int     localdummy=0;


028     sysinfo(&guestInfo);
029     printf("ZCZC RAM     %5lu MB\n",  guestInfo.totalram/1024/1024);
030     printf("ZCZC FREE    %5lu MB\n",  guestInfo.freeram/1024/1024);
031     printf("ZCZC BUFFER  %5lu MB\n",  guestInfo.bufferram/1024/1024);
032     printf("ZCZC SWAP    %5lu MB\n",  guestInfo.totalswap/1024/1024);
033     printf("ZCZC FREESW  %5lu MB\n",  guestInfo.freeswap/1024/1024);
034     IntPtr  intArray=malloc((ArraySize+1) * sizeof(int));
035     sysinfo(&guestInfo);
036     printf("ZCZC FREE1   %5lu MB\n",  guestInfo.freeram/1024/1024);
037     printf("ZCZC FREESW1 %5lu MB\n",  guestInfo.freeswap/1024/1024);
038     for (int ii=0; ii<ArraySize; ii++) intArray[ii]=255;
039     sysinfo(&guestInfo);
040     printf("ZCZC FREE2   %5lu MB\n",  guestInfo.freeram/1024/1024);
041     printf("ZCZC FREESW2 %5lu MB\n",  guestInfo.freeswap/1024/1024);
042     printMyAddress( printMyAddress, "printMyAddress()");
043     printMyAddress( main,           "main()");
044     printMyAddress(&pcounter,      "&pcounter");
045     printMyAddress( intArray,       "intArray");
046     printMyAddress( printf,         "printf()");
047     printMyAddress(&intArray,      "&intArray");
048     printMyAddress(&localdummy,    "&localdummy");
049     printMyAddress(&guestInfo,     "&guestInfo");
050     sleep(1);
051 }
```

(e) (77%) Variable "`pcounter`" is a 32-bit **LITTLE ENDIAN** variable. Is "`pcounter`" a local or global variable?

Variable "`pcounter`": _____

(f) (38%) What is the value of variable "`pcounter`" just before the program exits?

Value of "`pcounter`": _____

(g) (77%) What is the Virtual Address of "`pcounter`" (in **hexadecimal**)?

Virtual Address of "`pcounter`": _____ (HEX)

(h) (30%) Write down the value of "`pcounter`" into the virtual address of "`pcounter`" (in **hexadecimal**)!

Virtual Addresses (HEX)	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F