COMPUTER PROGRAMMING I
GROUP ( ) LAB (5)
ANSWERS TO VERBAL QUESTIONS

BY

NAME:

Student ID:

DATE: 09.12.2022

LAB INSTRUCTION

- LAB Mode: INDIVIDUAL .
- Subm tıon:-
  - Hard copy.
  - Softcopyas PDF format to Google Classroom.
  - ProgramıngCode.
- The font must be ARIAL andthe s ze must be 12 . JUSTIFY. SPACING 1.5.
- Use YOUR NAME, SURNAME , and DATE as a var able n youranswer. • Complete thecoversheetandattach t toyourlab as (f rstpage).
- Answerlabw thownwork (NO PLAGIARISM).
- Yourmarksw ll be deducted n thecase of:
  - Latesubm ss on. - Plag ar sm.

LAB QUESTIONS

1 . Answer each of the follow ng. Assume that s ngle-prec s on float ng-po nt numbers are stored n 4 bytes, and that the start ng address of the array s at locat on 1002500 n memory. Each part of the exerc se should use the results of prev ous parts where appropr ate.

a) Define an array of type float called numbers w th 10 elements, and n t al ze the elements to the values 0.0, 1.1, 2.2, …, 9.9. Assume the symbol c constant SIZE has been def ned as 10.

b) Def ne a pointer, nPtr, that po nts to an object of type float.

c) Pr nt the elements of array numbers us ng array ndex notat on. Use a for statement. Pr nt each number w th 1 pos t on of prec s on to the r ght of the dec mal po nt. d) G ve two separate statements that assign the start ng address of array numbers to the po nter var able nPtr.

e) Pr nt the elements of array numbers us ng po nter/offset notat on w th the po nter nPtr. f) Pr nt the elements of array numbers us ng po nter/offset notat on w th the array name as the po nter.

g) Pr nt the elements of array numbers by ndex ng po nter nPtr.

h) Assum ng that nPtr po nts to the beg nn ng of array numbers, what address s referenced by nPtr + 8? What value s stored at that locat on?

) Refer to element 4 of array numbers us ng array ndex notat on, po nter/offset notat on w th the array name as the po nter, po nter ndex notat on w th nPtr and po nter/offset notat on w th nPtr.

2. F nd the error n each of the follow ng program segments. Assume

```
int *zPtr; // zPtr will reference array z
int *aPtr = NULL;
void *sPtr = NULL;
int number;
int z[ 5] = { 1 , 2 , 3 , 4
, 5} ; sPtr = z;
```

a) ++zptr;

b) // use po nter to get f rst value of array;
assume zPtr s n t al zed number = zPtr;

c) // ass gn array element 2 (the value 3) to
number; assume zPtr s n t al zed

d) // pr nt ent re array z; assume zPtr s n t al zed

```
for ( size_t i = 0 ; i <= 5 ; ++i) {
printf( "%d " , zPtr[i]);
}
```

a) Define an array of type float called numbers w th 10 elements, and n t al ze the elements to the values 0.0, 1.1, 2.2, ..., 9.9. Assume the symbol c constant SIZE has been defined as 10.

Answer: The array of type float called numbers can be defined and initialized as follows:

float numbers[SIZE] = {0.0, 1.1, 2.2, 3.3, 4.4, 5.5, 6.6, 7.7, 8.8, 9.9} ;

b) Define a pointer, nPtr, that points to an object of type float.

Answer: The pointer nPtr can be defined as follows:

```
float *nPtr;
```

c) Print the elements of array numbers us ng array index notation. Use a for statement.

Print each number with 1 position of precision to the right of the decimal point.

Answer: The elements of the array numbers can be pr nted us ng a for loop and array index notation as follows:

```
for (size_t i = 0; i < SIZE; ++i) {

        printf("%.1f ", numbers[i]); }
```

d) Give two separate statements that assign the starting address of array numbers to the pointer variable nPtr.

Answer: Two statements that can assign the starting address of array numbers to the pointer variable nPtr are: nPtr = numbers; nPtr = &numbers[0];

e) Print the elements of array numbers using pointer/offset notation with the pointer nPtr.

Answer: The elements of the array numbers can be printed using pointer/offset notation with the pointer nPtr as follows:

```
for (size_t i = 0; i < SIZE; ++i) {

        printf("%.1f ", *(nPtr + i));}
```

f) Print the elements of array numbers using pointer/offset notation with the array name as the pointer.

Answer: The elements of the array numbers can be printed using pointer/offset notation with the array name as the pointer as follows:

```
for (size_t i = 0; i < SIZE; ++i) {

        printf("%.1f ", *(numbers + i));}
```

g) Print the elements of array numbers by indexing pointer nPtr.

Answer: The elements of the array numbers can be printed by indexing pointer nPtr as follows:

for (size_t i = 0; i < SIZE; ++i) {

printf("%.1f ", nPtr[i]); }

h) Assuming that nPtr points to the beginning of array numbers, what address is referenced by nPtr + 8? What value is stored at that location?

Answer: If nPtr points to the beginning of the array numbers, then nPtr + 8 would reference the address 1002532 in memory. The value stored at that location would be 8.8. The address is 1002500 + 8 * 4 = 1002532. The value is 8.8.

i) Refer to element 4 of array numbers using array index notation, pointer/offset notation with the array name as the pointer, pointer index notation with nPtr and pointer/offset notation with nPtr.

Answer: Element 4 of the array numbers can be referred to using the following notations:

· Array index notation: numbers[4]

· Pointer/offset notation with the array name as the pointer: *(numbers + 4) ·

Pointer ndex notation with nPtr: nPtr[4]

· Pointer/offset notation with nPtr: *(nPtr + 4)

2. Find the error n each of the follow ng program segments. Assume

int *zPtr; // zPtr will reference array z int *aPtr = NULL; void *sPtr =

NULL; int number; int z[5] = {1, 2, 3, 4, 5}; sPtr = z;

The errors n the g ven program segments are as follows:

a) ++ zptr;

Answer: The correct syntax to increment a pointer is ++zPtr (note the uppercase "P"). Initialize zPtr with zPtr = z; before performing the pointer arithmetic.

b) // use pointer to get first value of array; assume zPtr is initialized number = zPtr;

Answer: The statement number = zPtr; tries to assign the pointer variable zPtr to the integer variable number, which is not allowed. To get the first value of the array, the correct statement would be number = *zPtr;

c) // assign array element 2 (the value 3) to number; assume zPtr is initialized

Answer: Error: zPtr[2] is not a pointer and should not be dereferenced. Change *zPtr[2] to zPtr[2].

d) // print entire array z; assume zPtr is

initialized for (size_t i = 0; i <= 5; ++i) {

printf("%d ", zPtr[i]); }

Answer: The for loop in this program segment uses an incorrect loop condition. The condition <= 5 will cause the loop to run one extra time, resulting in an out-of bounds access to the array z. To correctly print the entire array z, the loop condition should be < 5. The correct for loop would be:

```
for (size_t i = 0; i < 5; ++i) {

    printf("%d ", zPtr[i]);

}
```

Change the operator <= in the for condition to <.