Tshwane University of Technology
We empower people

INSTRUCTIONS TO CANDIDATES

1. All exam rules stated by the Tshwane University of Technology apply.
2. **Ensure a single final version of your source code is handed in as requested.**
3. If needed, state all necessary assumptions clearly in code commentary.

MARKS: 100%

PAGES: 15 (incl. cover)

EXAMINER:

Mr A.J. Smith

Mr D. Engelbrecht

Prof J.A. Jordaan

MODERATOR:

Mr TE Olivier

TIME:

90 Minutes

15 Minutes extra time in the event of computer problems

FACULTY OF ENGINEERING AND THE BUILT ENVIRONMENT

DEPARTMENT OF ELECTRICAL ENGINEERING

ES216BB ENGINEERING SOFTWARE DESIGN B

EVALUATION 3

OCTOBER 2025

EVALUATION INSTRUCTIONS

1. **Plagiarism:** Submit only original work. We will use similarity software to verify the authenticity of all submissions.

2. **Permitted Tools:** You are allowed to use only **CodeBlocks** and **Google Chrome** to access the evaluation, view the evaluation PDF and upload your submission for this evaluation. Access to emails, other online resources, and memory sticks is strictly prohibited. Please be aware that computer activity will be remotely monitored. Breaches of TUT's official examination and module rules will result in a minimum penalty of zero for this evaluation, with the potential for further disciplinary action.

3. **File Submission:** Your header code file must be named according to this format: **"<student number>.h" (e.g. 21011022.h).** Do not add any other text (name, surname, etc.) to the file name (ONLY YOUR STUDENT NUMBER).

4. **Uploading Instructions:** **ONLY SUBMIT YOUR HEADER FILE (.h)** via the designated upload link. While multiple uploads are allowed, only the most recent submission will be retained on the system. If you make an error in your initial upload, simply re-upload your file, and the previous version will be overridden.

5. **Evaluation Scope:** This assessment encompasses basic content from ES216AB and specifically ES216BB content defined in **Units 1 to 5**

6. **Programming Language:** Construct your program in **C++** and adhere to structured programming principles.

7. **Editing and Requirements:** Your program must meet all specified requirements. Refer to the attached appendices for additional details.

8. **Evaluation Requirements:**

    a. Remember to save your work on the PC "D: Drive" and save regularly throughout the evaluation.
    b. Do not modify the given code in the ".cpp" file except for changing the header file name to your student number, e.g. "123456789.h".
    c. Do not modify the given libraries and comments in the template ".h" file.
    d. Complete the C++ class definition and functions in each comment block as shown in the template ".h" file.Use the exact function names and parameters as used in the evaluation question paper and given ".cpp" file.

C++ FILE CODE EXPLANATION

The provided C++ file sets up a program that manages a university course enrollment system. It uses a menu interface interacting with the **CourseEnrollment** class, which allows users to view and manage student enrollments for a specific course.

The system includes functionalities for:

- **Displaying Course Details**: Shows the course code, course title, maximum capacity, and number of students enrolled.
- **Loading Student Enrollments**: Reads student names from an external text file and adds them to the enrollment list.
- **Displaying Student List**: Prints all names of enrolled students.
- **Displaying Tuition Revenue**: Calculates total revenue based on student count and tuition fee per student.

This system relies on the **CourseEnrollment** class to encapsulate course information, manage dynamic memory for enrolled students, and compute tuition revenue.

The required class definition and functions hereafter for the C++ header file should be implemented in the appropriate comment blocks as given in the .h template file.

CLASS DEFINITION EXPLANATION

The **CourseEnrollment** class encapsulates the structure and behavior of a course registration system:

- **Private Members:**
  - **CourseCode** (**string**): The course identifier (e.g., CS101).
  - **CourseTitle** (**string**): The full title of the course.
  - **MaxStudents** (**int**): Maximum allowable enrollments.
  - **StudentList** (**string***): Dynamically allocated array to store enrolled student names.
  - **EnrolledCount** (**int**): Number of students currently enrolled.

- **Public Functions:**
  - **CourseEnrollment**(): Constructor.
  - **~CourseEnrollment**(): Destructor.
  - **SetDetails**(): Sets course code, title, and maximum enrollment.
  - **AddStudent**(): Adds a student to the course if space is available.
  - **DisplayCourseDetails**(): Shows course metadata.
  - **DisplayStudentList**(): Outputs all enrolled student names.
  - **TuitionRevenue**(): Computes total course revenue.

CLASS FUNCTION EXPLANATIONS

1. Constructor

CourseEnrollment::CourseEnrollment()

- **Purpose:** Initializes data members to safe defaults.
  - Strings are initialized empty.
  - Integers set to 0.
  - Pointer set to nullptr.
- **Parameters:** None.
- **Returns:** Nothing (constructor).

2. Destructor

CourseEnrollment::~CourseEnrollment()

- **Purpose:** Frees memory used by dynamically allocated StudentList.
- **Parameters:** None.
- **Returns:** Nothing (destructor).

3. SetDetails Function

void CourseEnrollment::SetDetails(string CC, string CT, int MS)

- **Purpose:** Assigns course code, title, and max student count.
- **Parameters:**
  - string CC – Course code.
  - string CT – Course title.
  - int MS – Maximum number of students.
- **Returns:** Nothing.

4. AddStudent Function

void CourseEnrollment::AddStudent(string StudentName)

- **Purpose:** Adds a student if the maximum capacity has not been reached.
- **Parameters:**
  - string StudentName – Name of the student.
- **Returns:** Nothing.

5. DisplayCourseDetails Function

void CourseEnrollment::DisplayCourseDetails()

- **Purpose:** Displays course code, title, maximum and current enrollment.
- **Parameters:** None.
- **Returns:** Nothing. Outputs to console.

6. DisplayStudentList Function

void CourseEnrollment::DisplayStudentList()

- **Purpose:** Outputs the names of all enrolled students.
    - Displays message if none are enrolled.
- **Parameters:** None.
- **Returns:** Nothing. Outputs to console.

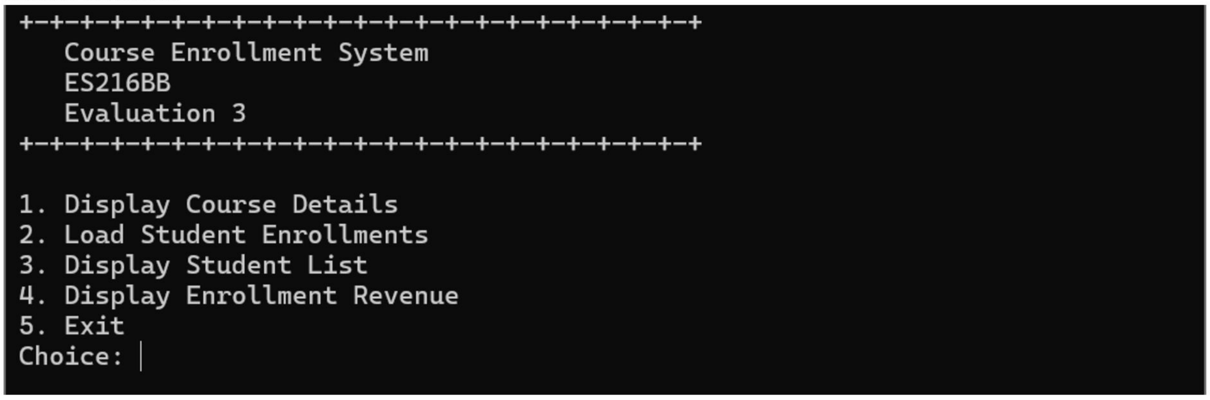
7. TuitionRevenue Function

float CourseEnrollment::TuitionRevenue(float FeePerStudent)

- **Purpose:** Calculates total income from enrolled students.
    - Computation: EnrolledCount * FeePerStudent
- **Parameters:**
    - float FeePerStudent – Tuition charged per student.
- **Returns:**
    - float value representing total revenue.

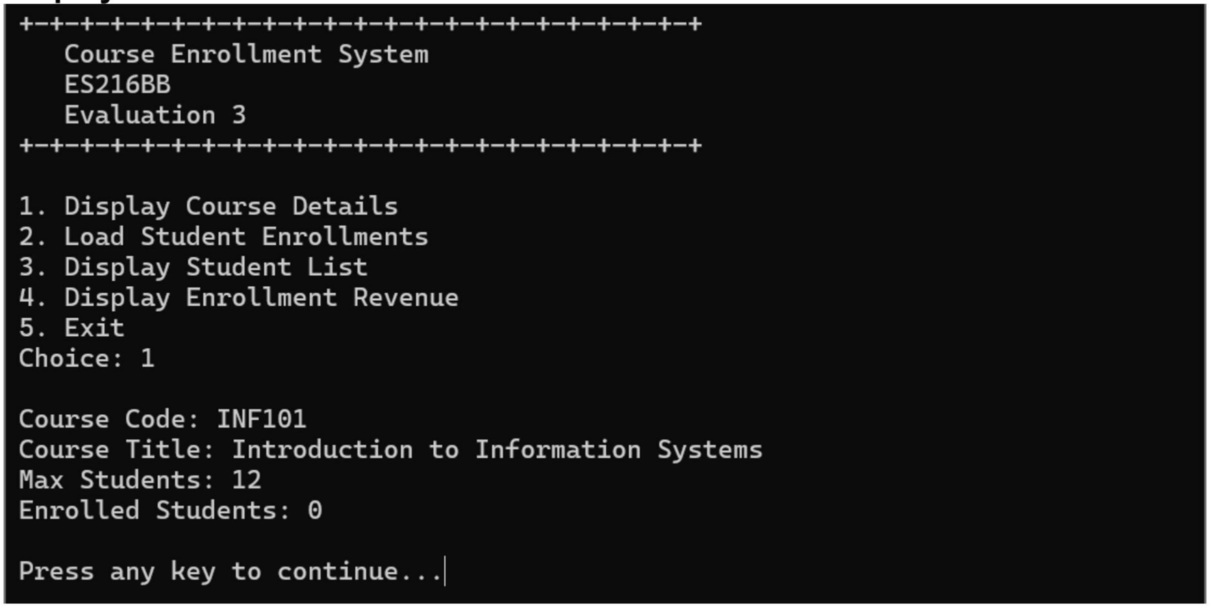
PRINT SCREENS

Main Menu:



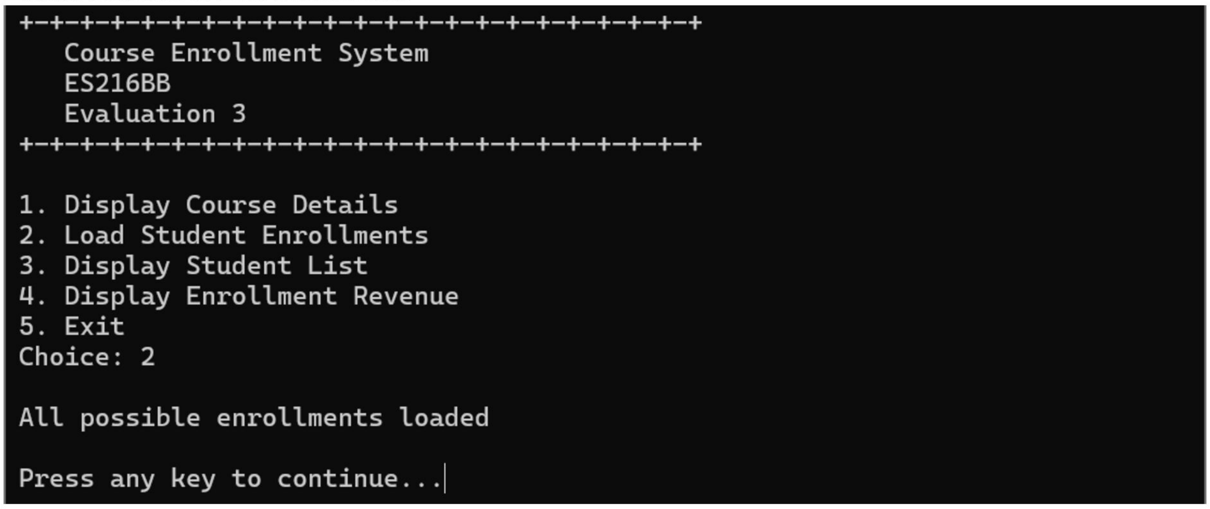
```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   Course Enrollment System
   ES216BB
   Evaluation 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

1. Display Course Details
2. Load Student Enrollments
3. Display Student List
4. Display Enrollment Revenue
5. Exit
Choice: 
```

Display Course Details:



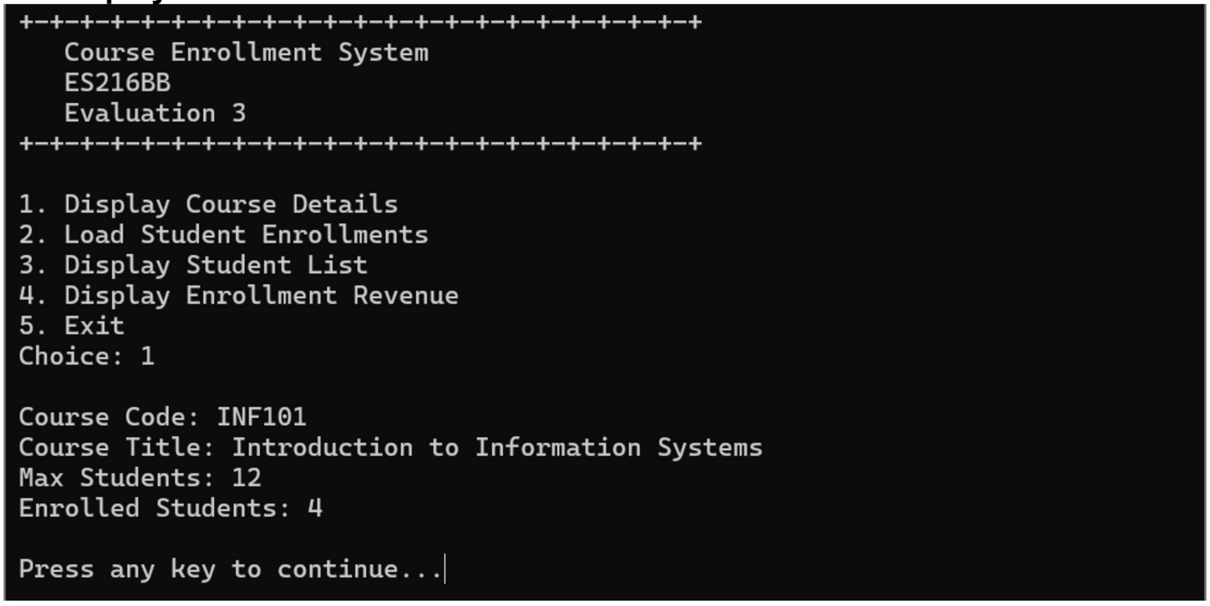
```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   Course Enrollment System
   ES216BB
   Evaluation 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

1. Display Course Details
2. Load Student Enrollments
3. Display Student List
4. Display Enrollment Revenue
5. Exit
Choice: 1

Course Code: INF101
Course Title: Introduction to Information Systems
Max Students: 12
Enrolled Students: 0

Press any key to continue...
```

Load Student Enrollments:



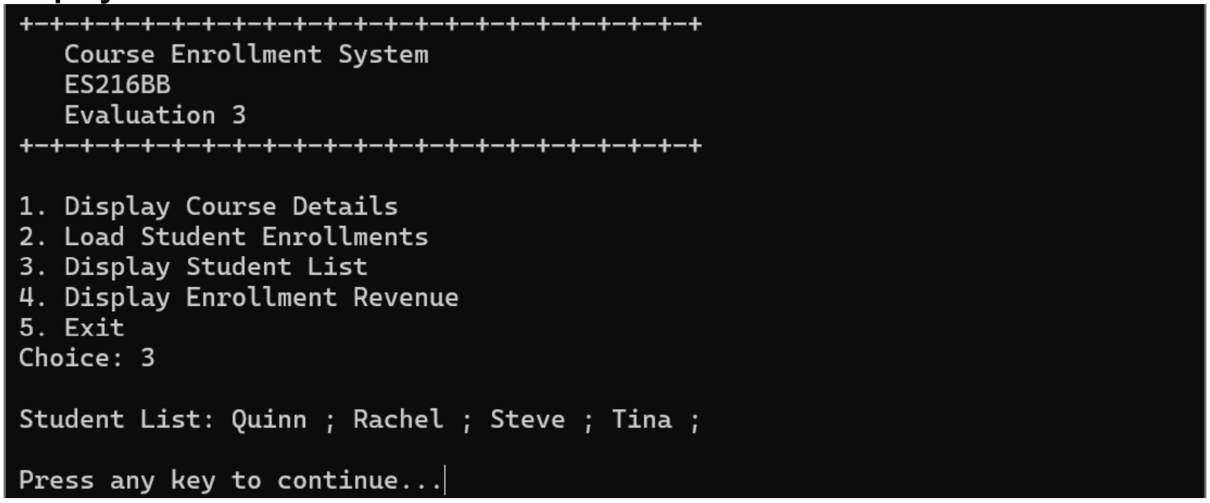
```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   Course Enrollment System
   ES216BB
   Evaluation 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

1. Display Course Details
2. Load Student Enrollments
3. Display Student List
4. Display Enrollment Revenue
5. Exit
Choice: 2

All possible enrollments loaded

Press any key to continue...
```

Re-Display Course Details:



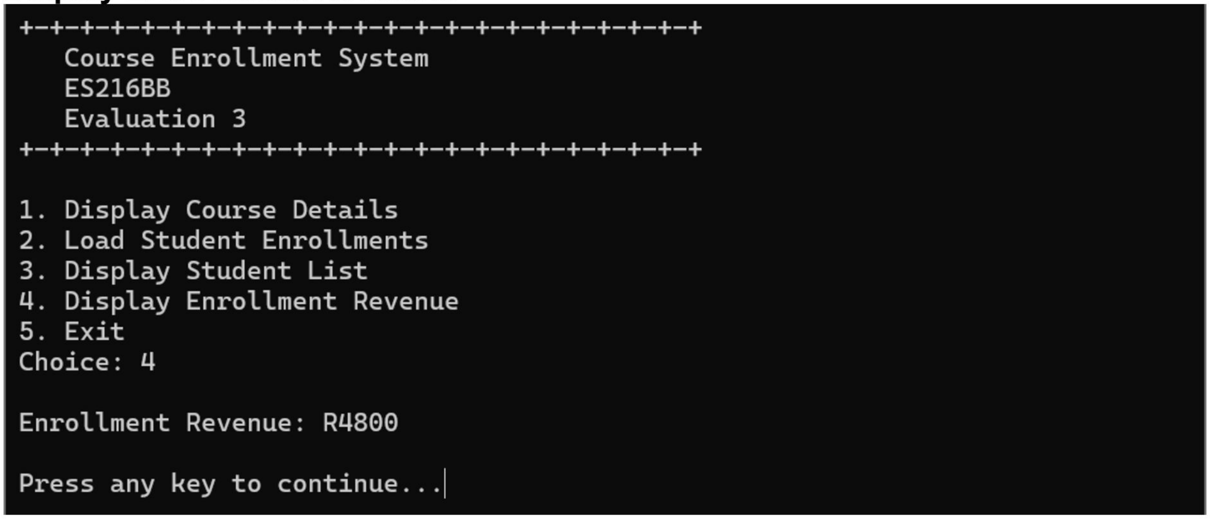
```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   Course Enrollment System
   ES216BB
   Evaluation 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

1. Display Course Details
2. Load Student Enrollments
3. Display Student List
4. Display Enrollment Revenue
5. Exit
Choice: 1

Course Code: INF101
Course Title: Introduction to Information Systems
Max Students: 12
Enrolled Students: 4

Press any key to continue...
```

Display Student List:



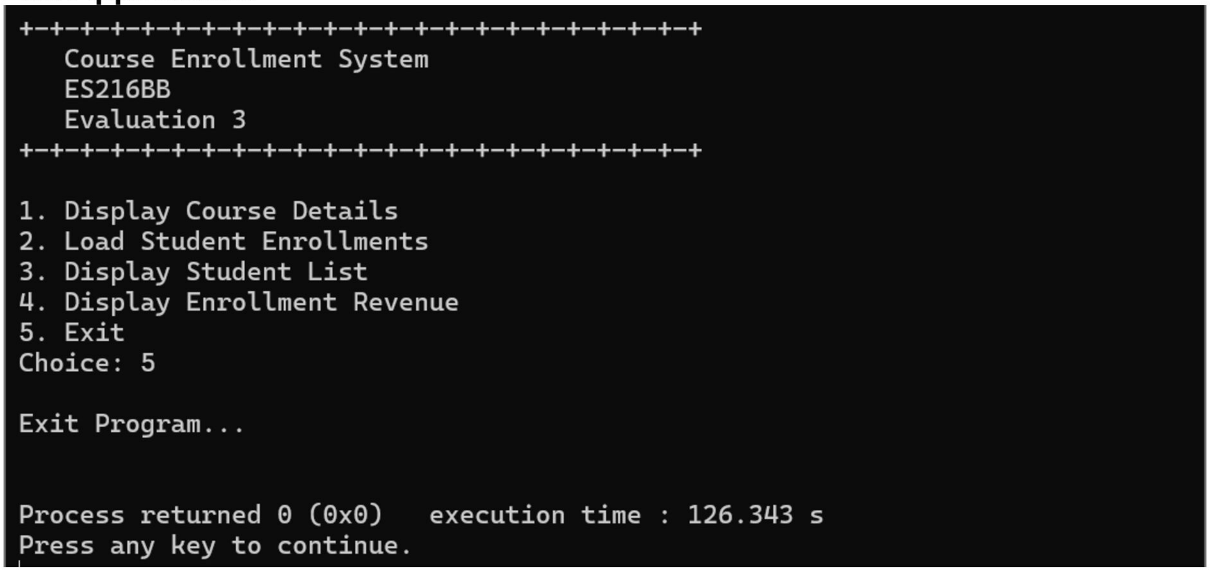
```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   Course Enrollment System
   ES216BB
   Evaluation 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

1. Display Course Details
2. Load Student Enrollments
3. Display Student List
4. Display Enrollment Revenue
5. Exit
Choice: 3

Student List: Quinn ; Rachel ; Steve ; Tina ;

Press any key to continue...
```

Display Entrollment Revenue:



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   Course Enrollment System
   ES216BB
   Evaluation 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

1. Display Course Details
2. Load Student Enrollments
3. Display Student List
4. Display Enrollment Revenue
5. Exit
Choice: 4

Enrollment Revenue: R4800

Press any key to continue...
```

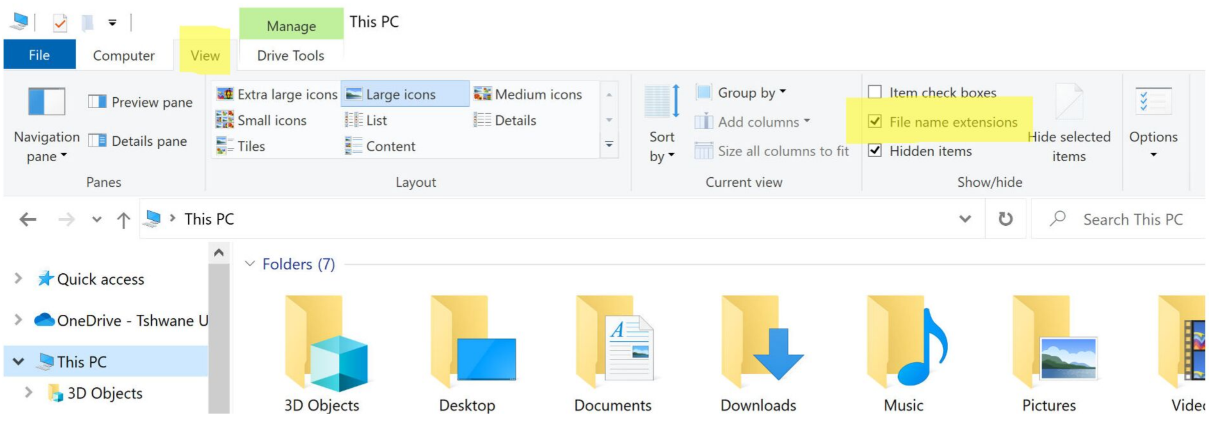
Exit Application:



```
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+
   Course Enrollment System
   ES216BB
   Evaluation 3
+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+-+

1. Display Course Details
2. Load Student Enrollments
3. Display Student List
4. Display Enrollment Revenue
5. Exit
Choice: 5

Exit Program...


Process returned 0 (0x0)   execution time : 126.343 s
Press any key to continue.
```

HOW TO RUN THE SHOWCASE FILE

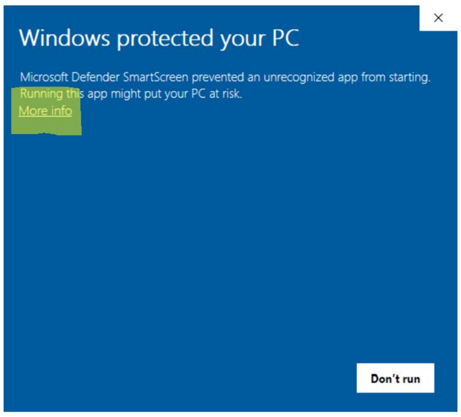
1. Enable file extensions (see highlighted in yellow)



2. Change the name from **"Showcase.old"** to **"Showcase.exe"**

3. Run the **"ShowcaseEV.exe"** by double-clicking on the icon.

4. Windows may show the following. Click on **"More info"**

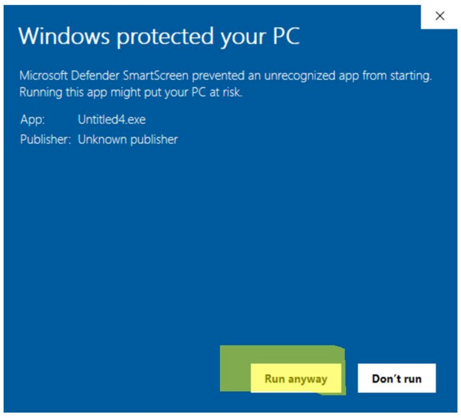


Windows protected your PC

Microsoft Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.

More info

5. Click on **"Run anyway"**



Windows protected your PC

Microsoft Defender SmartScreen prevented an unrecognized app from starting. Running this app might put your PC at risk.

App: Untitled4.exe
Publisher: Unknown publisher

ANNEXURE A – MARK ALLOCATION

Note: Score range is 0 - 4 which is: 0-none, 1-poor, 2-average, 3-good, 4-excellent

TEST RUBRIC	SCORE [0-4]	WEIGHT [%]
C++ CODE EVALUATION		**50+2**
0. Class Definition		**6**
1. Class Constructor Function		**4**
2. Class Destructor Function		**4**
3. Set Details Function		**5**
4. Add Function		**8**
5. Display Details Function		**5**
6. Display List Function		**5**
7. Profit Function		**5**
8. Overall Impression		**5**
9. No Compile or Runtime Errors		**5**
TOTAL		**50**

Graduate Attribute	GA Number	GA Score [0-5]
Application of scientific and engineering knowledge	GA2	4,7
Engineering methods, skills, tools, including information technology	GA5	0,1,2,3
Impact of Engineering Activity	GA7	5,6
Engineering Professionalism	GA10	8,9

ANNEXURE B – INFORMATION SHEET

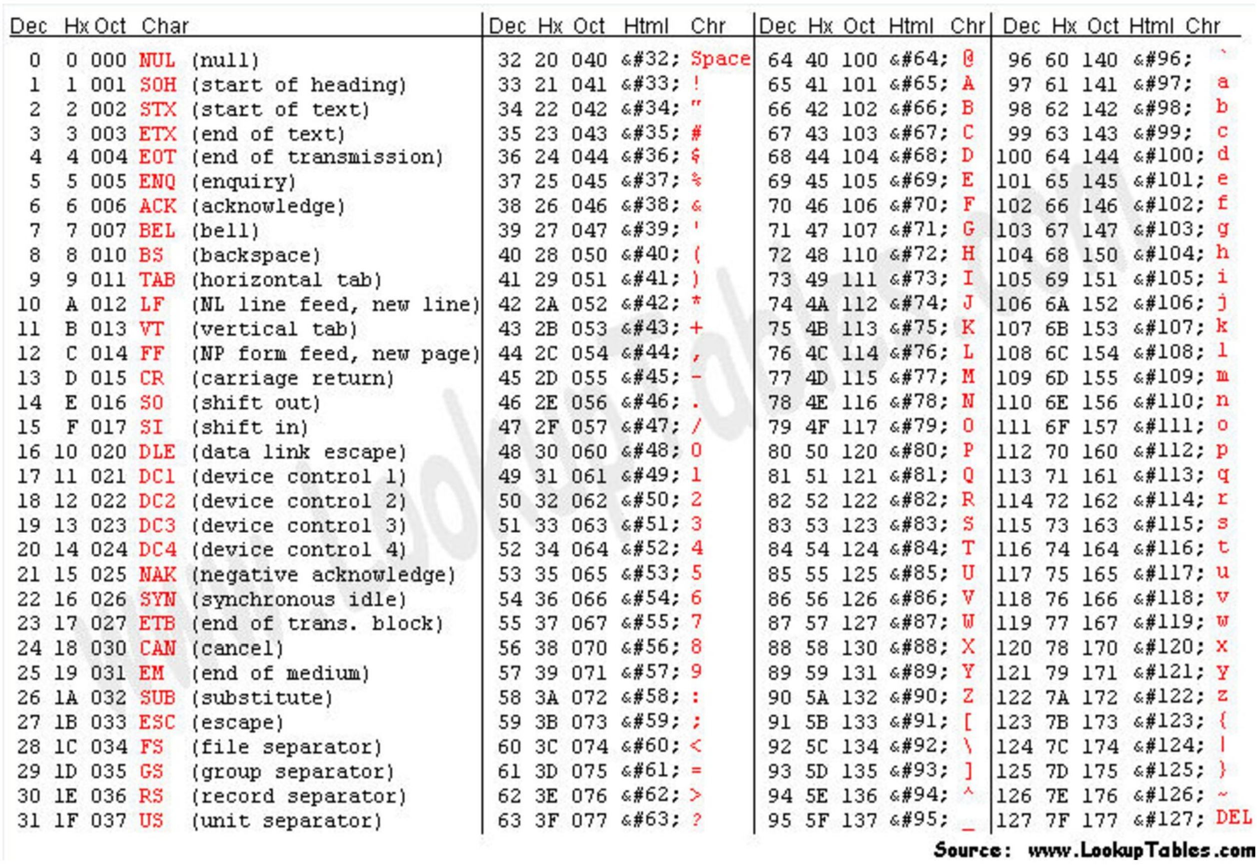
Data types: void, char, short, int, float, double
Data Type modifiers: const, auto, static, unsigned, signed
Arithmetic operators: * / % + -
Relational operators: < <= > >= == !=
Assignment operator: = += -= *= /= %= &= ^= |= <<= >>=
Logic operators: && || !
Bitwise logic operators: & | ^ ~ << >>
Pointer operators: Derefernce: * Address: &
Control Structures:

IF Selection:	if (condition) { … };
IF ELSE Selection:	if (condition) { … } else { … };
WHILE Loop:	while (condition) { … };
DO WHILE loop:	do { … } while (condition);
FOR Loop:	for (initial value of control variable; loop condition; increment of control variable) { … }
SWITCH Selection:	switch (control variable){ case ‘value’: … ; break; default: … ; break; }

Functions: return_data_type function_name (parameters) { … };
Common Library Functions: printf() , scanf() , rand() , srand() , time() , isalpha() , isdigit() , getchar() , getch(), strcpy()
Arrays:

One dimensional:	data_type variable_name[size];
Two dimensional:	data_type variable_name [x_size][y_size];

ANNEXURE C – ASCII TABLE



Dec	Hx	Oct	Char	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr	Dec	Hx	Oct	Html	Chr
0	0	000	NUL (null)	32	20	040	&#32;	Space	64	40	100	&#64;	@	96	60	140	&#96;	`
1	1	001	SOH (start of heading)	33	21	041	&#33;	!	65	41	101	&#65;	A	97	61	141	&#97;	a
2	2	002	STX (start of text)	34	22	042	&#34;	"	66	42	102	&#66;	B	98	62	142	&#98;	b
3	3	003	ETX (end of text)	35	23	043	&#35;	#	67	43	103	&#67;	C	99	63	143	&#99;	c
4	4	004	EOT (end of transmission)	36	24	044	&#36;	$	68	44	104	&#68;	D	100	64	144	&#100;	d
5	5	005	ENQ (enquiry)	37	25	045	&#37;	%	69	45	105	&#69;	E	101	65	145	&#101;	e
6	6	006	ACK (acknowledge)	38	26	046	&#38;	&	70	46	106	&#70;	F	102	66	146	&#102;	f
7	7	007	BEL (bell)	39	27	047	&#39;	'	71	47	107	&#71;	G	103	67	147	&#103;	g
8	8	010	BS (backspace)	40	28	050	&#40;	(	72	48	110	&#72;	H	104	68	150	&#104;	h
9	9	011	TAB (horizontal tab)	41	29	051	&#41;	)	73	49	111	&#73;	I	105	69	151	&#105;	i
10	A	012	LF (NL line feed, new line)	42	2A	052	&#42;	*	74	4A	112	&#74;	J	106	6A	152	&#106;	j
11	B	013	VT (vertical tab)	43	2B	053	&#43;	+	75	4B	113	&#75;	K	107	6B	153	&#107;	k
12	C	014	FF (NP form feed, new page)	44	2C	054	&#44;	,	76	4C	114	&#76;	L	108	6C	154	&#108;	l
13	D	015	CR (carriage return)	45	2D	055	&#45;	-	77	4D	115	&#77;	M	109	6D	155	&#109;	m
14	E	016	SO (shift out)	46	2E	056	&#46;	.	78	4E	116	&#78;	N	110	6E	156	&#110;	n
15	F	017	SI (shift in)	47	2F	057	&#47;	/	79	4F	117	&#79;	O	111	6F	157	&#111;	o
16	10	020	DLE (data link escape)	48	30	060	&#48;	0	80	50	120	&#80;	P	112	70	160	&#112;	p
17	11	021	DC1 (device control 1)	49	31	061	&#49;	1	81	51	121	&#81;	Q	113	71	161	&#113;	q
18	12	022	DC2 (device control 2)	50	32	062	&#50;	2	82	52	122	&#82;	R	114	72	162	&#114;	r
19	13	023	DC3 (device control 3)	51	33	063	&#51;	3	83	53	123	&#83;	S	115	73	163	&#115;	s
20	14	024	DC4 (device control 4)	52	34	064	&#52;	4	84	54	124	&#84;	T	116	74	164	&#116;	t
21	15	025	NAK (negative acknowledge)	53	35	065	&#53;	5	85	55	125	&#85;	U	117	75	165	&#117;	u
22	16	026	SYN (synchronous idle)	54	36	066	&#54;	6	86	56	126	&#86;	V	118	76	166	&#118;	v
23	17	027	ETB (end of trans. block)	55	37	067	&#55;	7	87	57	127	&#87;	W	119	77	167	&#119;	w
24	18	030	CAN (cancel)	56	38	070	&#56;	8	88	58	130	&#88;	X	120	78	170	&#120;	x
25	19	031	EM (end of medium)	57	39	071	&#57;	9	89	59	131	&#89;	Y	121	79	171	&#121;	y
26	1A	032	SUB (substitute)	58	3A	072	&#58;	:	90	5A	132	&#90;	Z	122	7A	172	&#122;	z
27	1B	033	ESC (escape)	59	3B	073	&#59;	;	91	5B	133	&#91;	[	123	7B	173	&#123;	{
28	1C	034	FS (file separator)	60	3C	074	&#60;	<	92	5C	134	&#92;	\	124	7C	174	&#124;	|
29	1D	035	GS (group separator)	61	3D	075	&#61;	=	93	5D	135	&#93;	]	125	7D	175	&#125;	}
30	1E	036	RS (record separator)	62	3E	076	&#62;	>	94	5E	136	&#94;	^	126	7E	176	&#126;	~
31	1F	037	US (unit separator)	63	3F	077	&#63;	?	95	5F	137	&#95;	_	127	7F	177	&#127;	DEL

Source: www.LookupTables.com