ICS 233 HOMEWORK02 KEY

Q01:

**includelib libcmt.lib**

**includelib libvcruntime.lib**

**includelib libucrt.lib**

**includelib legacy\_stdio\_definitions.lib**

**extern scanf\_s: proc, printf\_s: proc**

**.data**

 **promptMsg byte "Enter the three sides of a triangle in cm: ", 0**

 **inputFormatString byte "%lf %lf %lf", 0**

 **invalidInputMsg byte "Error: Invalid input.", 0**

 **nonPositiveInputMsg byte "Error: At least one of the inputs is <= 0.", 0**

 **notTriangleMsg byte "Error: The three sides do not form a triangle.", 0**

 **outputFormatString byte "The area of the triangle is %0.2f square cm.", 13, 10, 0**

 **TWO real8 2.0**

**.data?**

 **sideA real8 ?**

 **sideB real8 ?**

 **sideC real8 ?**

 **area real8 ?**

**.code**

**main proc**

 **push rbp**

 **mov rbp, rsp**

 **sub rsp, 32**

 **lea rcx, offset promptMsg ; printf("Enter the three sides of a triangle in cm: ")**

 **call printf\_s ;**

 **lea r9, sideC ; scanf("%lf %lf %lf", %sideA, %sideB, %sideC)**

 **lea r8, sideB ;**

 **lea rdx, sideA ;**

 **lea rcx, offset inputFormatString ;**

 **call scanf\_s ;**

 **cmp rax, 3 ; Valid input check**

 **jb ERROR\_MESSAGE ;**

 **cmp sideA, 0 ; is each side > 0 ?**

 **jb NON\_POSITIVE\_VALUE ;**

 **cmp sideB, 0 ;**

 **jb NON\_POSITIVE\_VALUE ;**

 **cmp sideC, 0 ;**

 **jb NON\_POSITIVE\_VALUE ;**

 **xorpd xmm0, xmm0 ; is a + b > c ?**

 **addsd xmm0, sideA ;**

 **addsd xmm0, sideB ;**

 **comisd xmm0, sideC ;**

 **jbe NOT\_TRIANGLE ;**

 **xorpd xmm0, xmm0 ; is a + c > b ?**

 **addsd xmm0, sideA ;**

 **addsd xmm0, sideC ;**

 **comisd xmm0, sideB ;**

 **jbe NOT\_TRIANGLE ;**

 **xorpd xmm0, xmm0 ; is b + c > a ?**

 **addsd xmm0, sideB ;**

 **addsd xmm0, sideC ;**

 **comisd xmm0, sideA ;**

 **jbe NOT\_TRIANGLE ;**

 **movsd xmm4, sideA ; XMM4 = (sideA + sideB + sideC) / 2**

 **addsd xmm4, sideB ;**

 **addsd xmm4, sideC ;**

 **divsd xmm4, TWO ;**

 **movsd xmm5, xmm4 ; XMM5 = (XMM4 - sideA)**

 **subsd xmm5, sideA ;**

 **movsd xmm6, xmm4 ; XMM6 = (XMM4 - sideB)**

 **subsd xmm6, sideB ;**

 **movsd xmm7, xmm4 ; XMM7 = (XMM4 - sideC)**

 **subsd xmm7, sideC ;**

 **mulsd xmm4, xmm5 ; xmm4 = xmm4 \* xmm5 \* xmm6 \* xmm7**

 **mulsd xmm4, xmm6 ;**

 **mulsd xmm4, xmm7 ;**

 **sqrtsd xmm4, xmm4 ; xmm4 = sqrt(xmm4)**

 **movsd area, xmm4 ;**

 **mov rdx, area ;**

 **lea rcx, outputFormatString ; printf("The area of the triangle is %0.2f square cm.")**

 **call printf\_s ;**

 **jmp \_END**

**ERROR\_MESSAGE:**

 **lea rcx, offset invalidInputMsg**

 **call printf\_s**

 **jmp \_END**

**NOT\_TRIANGLE:**

 **lea rcx, offset notTriangleMsg**

 **call printf\_s**

 **jmp \_END**

**NON\_POSITIVE\_VALUE:**

 **lea rcx, offset nonPositiveInputMsg**

 **call printf\_s**

**\_END:**

 **add rsp, 32**

 **pop rbp**

 **xor rax, rax**

 **ret**

**main endp**

**end**

**-----------------------------------------------------------------------------------------------------**

Q02:

**.686**

**.model flat, c**

**.XMM**

**includelib libcmt.lib**

**includelib libvcruntime.lib**

**includelib libucrt.lib**

**includelib legacy\_stdio\_definitions.lib**

**extern printf\_s: proc, scanf\_s: proc, sin: proc**

**.data**

**promptMsg byte "Enter two sides of a triangle in cm and the angle between them in degrees: ", 13, 10, 0**

 **inputFormatString byte "%lf %lf %lf", 0**

 **invalidInputMsg byte "Error: Invalid input.", 13, 10, 0**

 **invalidLengthOrAngleMsg byte "Error: Invalid length or invalid angle.", 13, 10, 0**

 **outputFormatString byte "The area of the triangle is %0.2f square cm.", 13, 10, 0**

 **TWO real8 2.0**

 **ZERO real8 0.0**

 **ONEEIGHTY real8 180.0**

 **PI real8 3.141592653**

**.data?**

 **sideA real8 ?**

 **sideB real8 ?**

 **angle real8 ?**

 **sinAngle real8 ?**

**.code**

**main proc**

 **push ebp**

 **mov ebp, esp**

 **sub esp, 32**

 **push offset promptMsg**

 **call printf\_s**

 **add esp, 4**

 **push offset angle**

 **push offset sideA**

 **push offset sideB**

 **push offset inputFormatString**

 **call scanf\_s**

 **add esp, 16**

 **cmp eax, 3 ; Valid input check**

 **jb INVALID\_INPUT ;**

 **movsd xmm4, sideA ; Positive values check**

 **comisd xmm4, ZERO ; for triangle sides**

 **jb INVALID\_LENGTH\_OR\_ANGLE ;**

 **movsd xmm4, sideB ;**

 **comisd xmm4, ZERO ;**

 **jb INVALID\_LENGTH\_OR\_ANGLE ;**

 **movsd xmm4, angle ; 0 <= Angle <= 180**

 **comisd xmm4, ZERO ;**

 **jbe INVALID\_LENGTH\_OR\_ANGLE ;**

 **comisd xmm4, ONEEIGHTY ;**

 **jae INVALID\_LENGTH\_OR\_ANGLE ;**

 **movsd xmm5, angle ; convert angle to radians**

 **mulsd xmm5, PI ;**

 **divsd xmm5, ONEEIGHTY ;**

 **sub esp, 8 ; place value in lower 64-bits of xmm5 at top of stack**

 **movsd real8 ptr [esp], xmm5 ;**

 **call sin ; value returned in ST0**

 **add esp, 8 ; clear the stack**

 **fstp real8 ptr sinAngle ; sinAngle <--- ST0**

 **movsd xmm7, sideA ; calculate area**

 **mulsd xmm7, sideB ;**

 **mulsd xmm7, sinAngle ;**

 **divsd xmm7, TWO ;**

 **sub esp, 8 ; display the area**

 **movsd real8 ptr [esp], xmm7 ;**

 **push offset outputFormatString ;**

 **call printf\_s ;**

 **add esp, 12 ;**

 **jmp \_END**

**INVALID\_LENGTH\_OR\_ANGLE:**

 **push offset invalidLengthOrAngleMsg**

 **call printf\_s**

 **add esp, 4**

 **jmp \_END**

**INVALID\_INPUT:**

 **push offset invalidInputMsg**

 **call printf\_s**

 **add esp, 4**

**\_END:**

 **add esp, 32**

 **pop ebp**

 **xor eax, eax**

 **ret**

**main endp**

**end**

**-----------------------------------------------------------------------------------------------------**

Q03:

**includelib libcmt.lib**

**includelib libvcruntime.lib**

**includelib libucrt.lib**

**includelib legacy\_stdio\_definitions.lib**

**extern printf\_s: proc, scanf\_s:proc**

**.data?**

 **value dword ?**

 **min dword ?**

 **N dword ?**

 **frequencyMin dword ?**

 **k dword ?**

**.data**

 **sumNegativeValues dword 0**

 **countNegativeValues dword 0**

 **promptMsg1 byte "Enter the number of values to consider:", 13, 10, 0**

 **promptMsg2 byte "Enter a value: ", 0**

 **invalidInputMsg byte "Error: Invalid input.", 13, 10, 0**

 **NoNegativeValueMsg byte "No negative values entered.", 13, 10, 0**

 **outputFormatstring1 byte "The smallest value is %d", 13, 10, 0**

 **outputFormatString2 byte "The frequency of the smallest value is %d", 13, 10, 0**

 **outputFormatString3 byte "The average of negative values is %0.2f", 13, 10, 0**

 **inputFormatString byte "%d", 0**

**.code**

**main proc**

 **enter 32, 0**

 **lea rcx, promptMsg1 ; printf\_s("Enter the number of values to consider:\n");**

 **call printf\_s ;**

 **lea rdx, N ; scanf\_s("%d", &N);**

 **lea rcx, inputFormatString ;**

 **call scanf\_s ;**

 **cmp N , 0 ;is N <= 0 ?**

 **jg L1 ; if No, jump to L1**

 **lea rcx, invalidInputMsg ;printf\_s("Error: Invalid input.\n");**

 **call printf\_s ;**

 **jmp L9**

**L1: lea rcx,promptMsg2 ; printf\_s("Enter a value: ");**

 **call printf\_s ;**

 **lea rdx, value ; scanf\_s("%d", &value);**

 **lea rcx, inputFormatString ;**

 **call scanf\_s ;**

 **mov eax, value ; min = value;**

 **mov min, eax ;**

 **mov frequencyMin, 1 ; frequencyMin = 1;**

 **cmp value, 0 ; is value < 0 ?**

 **jge L2 ; if No, jmp to L2**

 **inc countNegativeValues ; countNegativeValues++;**

 **mov eax, value ; sumNegativeValues += value;**

 **add sumNegativeValues, eax ;**

**L2: mov k, 2 ; for (k = 2; k <= N; k++)**

**L3: mov eax, N**

 **cmp k, eax**

 **jg L7**

 **lea rcx,promptMsg2 ; printf\_s("Enter a value: ");**

 **call printf\_s ;**

 **lea rdx, value ; scanf\_s("%d", &value);**

 **lea rcx, inputFormatString ;**

 **call scanf\_s ;**

 **cmp value, 0 ; is value < 0 ?**

 **jge L4 ; if No, jmp to L4**

 **inc countNegativeValues ; countNegativeValues++;**

 **mov eax, value ; sumNegativeValues += value;**

 **add sumNegativeValues , eax ;**

**L4: mov eax, min**

 **cmp value, eax ; is value == min ?**

 **jne L5 ; if No, jump to L5**

 **inc frequencyMin ; frequencyMin++;**

 **jmp L6**

**L5: mov eax, min**

 **cmp value, eax ; is value < min ?**

 **jge L6 ; if No, jump to L6**

 **mov eax, value ; min = value;**

 **mov min, eax ;**

 **mov frequencyMin, 1 ; frequency = 1;**

 **L6: inc k**

 **jmp L3**

**L7: mov edx, min ; printf\_s("The smallest value is %d\n", min);**

 **lea rcx, outputFormatstring1 ;**

 **call printf\_s ;**

 **mov edx, frequencyMin ; printf\_s("The frequency of the smallest value is %d\n", frequencyMin);**

 **lea rcx, outputFormatString2 ;**

 **call printf\_s ;**

 **cmp countNegativeValues, 0 ; is countNegativeValues == 0 ?**

 **jne L8 ; if no, jump to L8**

 **lea rcx,NoNegativeValueMsg ; printf\_s("No negative values entered.\n");**

 **call printf\_s ;**

 **jmp L9 ;**

 **; printf\_s("The average of negative values is %0.2f\n", (double)sumNegativeValues / count);**

**L8: cvtsi2sd xmm0,dword ptr sumNegativeValues ;**

 **cvtsi2sd xmm1,dword ptr countNegativeValues ;**

 **divsd xmm0,xmm1 ;**

 **movq rdx,xmm0 ;**

 **lea rcx, outputFormatString3 ;**

 **call printf\_s ;**

**L9: leave**

 **xor eax,eax ; return 0;**

 **ret ;**

**main endp**

**end**